

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of : THE COMMISSIONER IS AUTHORIZED
Kunihiko HAYASHI : TO CHARGE ANY DEFICIENCY IN THE
Serial No. NEW : FEES FOR THIS PAPER TO DEPOSIT
 : ACCOUNT NO. 23-0975
 : **Attn: APPLICATION BRANCH**
Filed March 10, 2004 : Attorney Docket No. 2004_0378A

TASK SWITCHING APPARATUS, METHOD
AND PROGRAM

CLAIM OF PRIORITY UNDER 35 USC 119

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450


Sir:

Applicant in the above-entitled application hereby claims the date of priority under the International Convention of Japanese Patent Application No. 2003-068831, filed March 13, 2003, as acknowledged in the Declaration of this application.

A certified copy of said Japanese Patent Application is submitted herewith.

Respectfully submitted,

Kunihiko HAYASHI

By 
Michael S. Huppert
Registration No. 40,268
Attorney for Applicant

MSH/kjf
Washington, D.C. 20006-1021
Telephone (202) 721-8200
Facsimile (202) 721-8250
March 10, 2004

日本国特許庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日 2003年 3月13日
Date of Application:

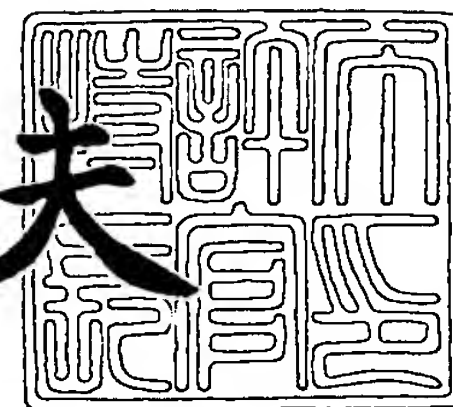
出願番号 特願2003-068831
Application Number:
[ST. 10/C]: [JP 2003-068831]

出願人 松下電器産業株式会社
Applicant(s):

2003年12月26日

特許庁長官
Commissioner,
Japan Patent Office

今井康夫



出証番号 出証特2003-3108030



【書類名】 特許願

【整理番号】 5037740132

【あて先】 特許庁長官殿

【国際特許分類】 G06F 9/46

【発明者】

【住所又は居所】 大阪府門真市大字門真 1 0 0 6 番地 松下電器産業株式会社内

【氏名】 林 ▲邦▼彦

【特許出願人】

【識別番号】 000005821

【氏名又は名称】 松下電器産業株式会社

【代理人】

【識別番号】 100109210

【弁理士】

【氏名又は名称】 新居 広守

【電話番号】 06-4806-7530

【手数料の表示】

【予納台帳番号】 049515

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 0213583

【プルーフの要否】 要



【書類名】 明細書

【発明の名称】 タスク切換装置、方法及びプログラム

【特許請求の範囲】

【請求項 1】 プロセッサにおいてタイムスロットを切り換えることによりタイムスロットに割り当てられたタスクの実行を切り換えるタスク切換装置であって、

第 1 タイプの複数のタスクのそれぞれを 1 つのタイムスロットに割り当て、第 1 タイプとは異なる第 2 タイプの複数のタスクを 1 つの特定のタイムスロットに割り当てる割当手段と、

前記特定のタイムスロット以外のタイムスロットに切り換えられた場合に、当該タイムスロットに割り当てられたタスクを選択し、前記特定のタイムスロットに切り換えられた場合に、第 2 タイプの複数のタスクから 1 つのタスクを選択するタスク選択手段と、

を備えることを特徴とするタスク切換装置。

【請求項 2】 前記第 2 タイプのタスクは優先度を有し、

前記タスク選択手段は、第 2 タイプの複数のタスクのうち優先度に従って 1 つのタスクを選択する

ことを特徴とする請求項 1 記載のタスク切換装置。

【請求項 3】 前記割当手段は、予め定められた一周期の時間から第 1 タイプのタスクが割り当てられたタイムスロットの合計時間を引いた残り時間を前記特定のタイムスロットの時間とする

ことを特徴とする請求項 1 又は 2 記載のタスク切換装置。

【請求項 4】 前記割当手段は、第 1 タイプの新たなタスクをタイムスロットを割り当てる毎に、前記残り時間を再計算して前記特定のタイムスロットの時間とする

ことを特徴とする請求項 3 記載のタスク切換装置。

【請求項 5】 前記第 1 タイプのタスクは割当時間の指定を有するタスクであり、

前記割当手段は、第 1 タイプの新たなタスクを追加しようとする場合、既に割



当済のタスクの割当時間の合計と新たなタスクの割当時間との総和が一周期の時間を超えていれば、当該新たなタスクのタイムスロット割り当てを拒否することを特徴とする請求項 1 記載のタスク切換装置。

【請求項 6】 前記タスク切換装置は、さらに
タスクによりアクセス可能な資源について、何れかのタスクからのアクセスによって資源がロック状態にあるか否かを示すロック情報を記憶する記憶手段と、
実行中のタスクがロック状態の資源をアクセスしようとしたとき、当該タスクを実行可能状態から待ち状態に変更し、当該資源のロック状態が解除されたとき待ち状態のタスクを実行可能状態に変更する変更手段と、
を備え、
前記タスク選択手段は、待ち状態にあるタスクを選択対象から除外することを特徴とする請求項 1 記載のタスク切換装置。

【請求項 7】 前記タスク切換装置は、さらに、
第 1 タイプ及び第 2 タイプのタスクのうち実行可能状態のタスクが 1 つも存在しない場合に、前記プロセッサを省電力状態に移行させる移行手段を備えることを特徴とする請求項 6 記載のタスク切換装置。

【請求項 8】 前記プロセッサは、タスクのコンテキストを保持するためのレジスタセットを少なくとも 2 つ備え、
前記タスク切換装置は、さらに、
レジスタセットの 1 つを実行中のタスクの使用に供し、他の 1 つのレジスタセットに対して次に実行すべきタスクのコンテキストをバックグラウンド処理により復帰させ、タイムスロットの切換時にレジスタセットを切り換える切換手段を備える

こと特徴とする請求項 1 から 7 の何れかに記載のタスク切換装置。

【請求項 9】 タスクが割り当てられたタイムスロットを切り換えることによりプロセッサにおいて実行すべきタスクを切り換えるタスク切換装置であって、

割当時間の指定を有する第 1 タイプのタスクに対して 1 つのタスクに 1 つのタイムスロットを割り当て、当該タイムスロット毎に対応するタスクの割当時間を



含むタイムスロット情報を生成する第1生成手段と、

優先度を有する第2タイプのタスクに対して複数のタスクを特定の1つのタイムスロットに割り当て、特定のタイムスロットの割当時間と優先度とを含む1つのタイムスロット情報を生成する第2生成手段と、

タイムスロットに割り当てられたタスク毎にタスクをアドレスを含むタスク管理情報を生成する第3生成手段と、

生成されたタイムスロット情報とタスク管理情報とを対応させて記憶する記憶手段と、

記憶手段に記憶されたタイムスロット情報を一周期内で少なくとも1つずつ選択する選択手段と、

第1タイプのタスクが割り当てられているタイムスロット情報が選択された場合に、当該タイムスロット情報に対応するタスク管理情報が示すタスクを実行させ、第2タイプのタスクが割り当てられているタイムスロット情報が選択された場合に、当該タイムスロット情報に対応する複数のタスク管理情報から優先度に従って1つを選択して、選択したタスク管理情報が示すタスクを実行させる制御手段と、

を備えることを特徴とするタスク切換装置。

【請求項10】 前記記憶手段は、第2タイプのタスクのタスク管理情報を優先度の順に並べたキューとして記憶し、

前記制御手段は、キューの先頭のタスク管理情報に対応するタスクを選択することを特徴とする請求項9記載のタスク切換装置。

【請求項11】 前記第2生成手段は、前記周期と、第1タイプの全タスクの割当時間の合計との差分を、特定のタイムスロットの割当時間として前記特定のタイムスロット情報に設定する

ことを特徴とする請求項10記載のタスク切換装置。

【請求項12】 前記第2生成手段は、第1生成手段によって第1タイプの新たなタスクにタイムスロットが割り当てられる毎に、前記残り時間を再計算して前記特定のタイムスロットの割当時間とする

ことを特徴とする請求項11記載のタスク切換装置。

【請求項 13】 前記記憶手段は、さらに、タスクによりアクセス可能な資源について、何れかのタスクからのアクセスによるロック状態であるか否かを示すロック情報を記憶し、

前記タスク切換装置は、さらに、

実行中のタスクがロック状態の資源をアクセスしようとしたとき、記憶手段に記憶された当該タスクのタスク管理情報とタイムスロット情報との対応関係を切り離し、当該タスク管理情報を待ちキューとして記憶させ、当該資源がロック状態を解除されたとき待ちキュー内のタスク管理情報をタイムスロット情報に対応させて記憶させるキュー管理手段を有する

ことを特徴とする請求項 12 記載のタスク切換装置。

【請求項 14】 前記プロセッサは、タスクのコンテキストを保持するためのレジスタセットを少なくとも 2 つ備え、

前記タスク切換装置は、さらに、

レジスタセットの 1 つを実行中のタスクの使用に供し、他の 1 つのレジスタセットに対して次に実行すべきタスクのコンテキストをバックグラウンド処理により復帰させ、タイムスロットの切換時にレジスタセットを切り換えるレジスタセット切換手段を備える

ことを特徴とする請求項 13 記載のタスク切換装置。

【請求項 15】 プロセッサにおいてタイムスロットを切り換えることによりタイムスロットに割り当てられたタスクの実行を切り換えるタスク切換方法であって、

第 1 タイプの複数のタスクのそれぞれを 1 つのタイムスロットに割り当て、第 1 タイプとは異なる第 2 タイプの複数のタスクを 1 つの特定のタイムスロットに割り当てる割当ステップと、

前記特定のタイムスロット以外のタイムスロットに切り換えられた場合に、当該タイムスロットに割り当てられたタスクを選択し、前記特定のタイムスロットに切り換えられた場合に、第 2 タイプの複数のタスクから 1 つのタスクを選択するタスク選択ステップと、

を有することを特徴とするタスク切換方法。

【請求項 16】 プロセッサにおいてタイムスロットを切り換えることによりタイムスロットに割り当てられたタスクの実行を切り換えるためのプログラムであって、

前記プログラムは、

第 1 タイプの複数のタスクのそれぞれを 1 つのタイムスロットに割り当て、第 1 タイプとは異なる第 2 タイプの複数のタスクを 1 つの特定のタイムスロットに割り当てる割当ステップと、

切り換え後のタイムスロットが前記特定のタイムスロットでない場合、当該タイムスロットに割り当てられたタスクを選択し、切り換え後のタイムスロットが前記特定のタイムスロットである場合、第 2 タイプの複数のタスクから 1 つのタスクを選択して実行させるタスク選択ステップと、

を前記プロセッサに実行させることを特徴とするプログラム。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、オペレーティングシステムにおけるタスク切換に関し、特にタスクが割り当てられたタイムスロットを切り換えることによりプロセッサにおいて実行すべきタスクを切り換えるタスク切換装置に関する。

【0002】

【従来の技術】

オペレーティングシステムの主な機能は、ハードウェア管理、タスク管理、データ管理及び入出力管理である。なかでもタスク管理はタスクの実行順序を管理するものであり、CPU やメモリ、入出力装置等を効率良く動作させるための重要な機能である。

【0003】

タスクとは、プログラムの起動とその実行、終了などの流れを一括管理する制御単位のことである。オペレーティングシステムの管理下で動作するプログラムはタスクとして扱われ、プログラムの実行に関するオペレーティングシステムの操作はすべてタスクを単位として行われる。

【0004】

タスクの実行順序を決定するアルゴリズムの一つとして、時分割スケジューリング法や優先度による切換方法がある。時分割スケジューリング法は、タスクにある実行時間を割り当て、割り当てられた時間の間はプロセッサの実行権がタスクに与えられ、割り当てられた時間が過ぎると別のタスクに実行権を移す方法である。これによりすべてのタスクに平等かつ決められた時間で実行権が割り当てられる。

【0005】

また、優先度による切換方法に関して、特許文献1及び2はいずれもタスクの優先度に従ってタスクを切り換えるスケジューリング装置を開示している。

【0006】**【特許文献1】**

特開2000-20323号公報

【0007】**【特許文献2】**

特開平4-101233号公報

【0008】**【発明が解決しようとする課題】**

しかしながら、上記従来技術によれば、各タスクの必要性能を確保するためにはプログラマが個々のタスク優先度を指定する点でプログラム設計が難しく、また一度設計してしまうと変更の柔軟性が乏しいという問題がある。

【0009】

本発明では、上記問題を解決するために、各タスクの必要性能を満たすための優先度を指定するプログラム設計を容易にし、プログラム設計の柔軟性を有するタスク切換装置を提供することを目的とする。

【0010】**【課題を解決するための手段】**

上記の目的を達成するために、本発明のタスク切換装置は、プロセッサにおいてタイムスロットを切り換えることによりタイムスロットに割り当てられたタス

クの実行を切り換えるタスク切換装置であって、第1タイプの複数のタスクのそれぞれを1つのタイムスロットに割り当て、第1タイプとは異なる第2タイプの複数のタスクを1つの特定のタイムスロットに割り当てる割当手段と、切り換え後のタイムスロットが前記特定のタイムスロットでない場合、当該タイムスロットに割り当てられたタスクを選択し、

切り換え後のタイムスロットが前記特定のタイムスロットでない場合、当該タイムスロットに割り当てられたタスクを選択し、切り換え後のタイムスロットが前記特定のタイムスロットである場合、第2タイプの複数のタスクから1つのタスクを選択して実行させるタスク選択手段とを備える。

【0011】

この構成によれば、第1タイプのタスクは1対1でタイムスロットを割り当てられる結果、第1タイプのタスクについては全てのタイムスロットを含む一周期内で少なくとも1回は選択実行されるので、継続した処理性能が保証される。他方、第2タイプのタスクは多対1で1つのタイムスロットを割り当てられる結果、継続して処理性能が保証されない。その結果、プログラマは、継続して処理性能を満たす必要のあるタスクに対しては優先度を考慮する必要もなく第1タイプに分類するだけでよい。また、プログラマは、処理性能を要求しないタスクに対しては第2タイプに分類すればよい。これにより、全てのタスクに対して個別に優先度を付与する必要がなくなり、処理性能を確保するためのプログラム設計を容易にすることができ、プログラム設計の柔軟性を確保することができる。

【0012】

ここで、前記第2タイプのタスクは優先度を有し、前記タスク選択手段は第2タイプの複数のタスクのうち優先度に従って1つのタスクを選択する構成としてもよい。

【0013】

この構成によれば、第2タイプのタスクは優先度の高いものから実行されるので、プログラマは、第1タイプのタスクに対して優先度を考慮する必要がなく、継続した処理性能を要求しないタスクを第2タイプに分類して優先度を付与すればよいので、プログラム設計及び変更を容易にすることができる。

【0014】

また、前記割当手段は、予め定められた一周期の時間から第1タイプのタスクが割り当てられたタイムスロットの合計時間を引いた残り時間を前記特定のタイムスロットの時間とするように構成してもよい。

【0015】

この構成によれば、第2タイプのタスクは上記の残り時間に実行されるので、第2タイプのタスクの実行により第1タイプの個々のタスクの処理性能に影響を及ぼすことを極力排除することができる。

【0016】

さらに、前記割当手段は、第1タイプの新たなタスクをタイムスロットを割り当てる毎に、前記残り時間を再計算して前記特定のタイムスロットの時間とする構成としてもよい。

【0017】

この構成によれば、第1タイプの個々のタスクの処理性能を保証した上で、第2タイプのタスクに対して最大限の残り時間を動的に割り当てることができる。

また、前記第1タイプのタスクは割当時間の指定を有するタスクであり、前記割当手段は、第1タイプの新たなタスクを追加しようとする場合、既に割当済のタスクの割当時間の合計と新たなタスクの割当時間との総和が一周期の時間を超えていれば、当該新たなタスクのタイムスロット割り当てを拒否する構成としてもよい。

【0018】

この構成によれば、第1タイプの新たなタスクの追加を排除してまで既存の第1タイプのタスクの処理性能を保証することができる。

さらに、前記タスク切換装置は、タスクによりアクセス可能な資源について、何れかのタスクからのアクセスによって資源がロック状態にあるか否かを示すロック情報を記憶する記憶手段と、実行中のタスクがロック状態の資源をアクセスしようとしたとき、当該タスクを実行可能状態から待ち状態に変更し、当該資源のロック状態が解除されたとき待ち状態のタスクを実行可能状態に変更する変更手段とを備え、前記タスク選択手段は、待ち状態にあるタスクを選択対象から除外

する構成としてもよい。

【0019】

また、前記タスク切換装置は、さらに、第1タイプ及び第2タイプのタスクのうち実行可能状態のタスクが1つも存在しない場合に、前記プロセッサを省電力状態に移行させる移行手段を備える構成としてもよい。

【0020】

ここで、前記プロセッサは、タスクのコンテキストを保持するためのレジスタセットを少なくとも2つ備え、前記タスク切換装置は、さらに、レジスタセットの1つを実行中のタスクの使用に供し、他の1つのレジスタセットに対して次に実行すべきタスクのコンテキストをバックグラウンド処理により復帰させ、タイムスロットの切換時にレジスタセットを切り換える切換手段を備える構成としてもよい。

【0021】

この構成によれば、コンテキストの退避及び復帰をプログラムの実行する代わりにレジスタセットの切換を行えばよいので、タスク切換を高速化することができる。また、次に実行すべきタスクのコンテキストをバックグラウンド処理により復帰させるので、プロセッサ時間のアイドル処理などを効率よく利用してタスク切換の高速化を図ることができる。

【0022】

また、本発明のタスク切換方法およびそのプログラムについても、上記と同様の構成であり、同様の作用及び効果を有している。

【0023】

【発明の実施の形態】

〔全体構成〕

図1は本発明の実施の形態におけるタスク切換を行うプログラム実行装置の主要部の構成を示すブロック図である。同図は、プロセッサにおいてオペレーティングシステムの一部の機能としてタスク切換を行うソフトウェアを実行することにより実現される機能を模式的に表している。プロセッサのハードウェア構成は、一般的なものでもよい。

【0024】

このプログラム実行装置は、割当時間の指定を有するタスクと優先度を有するタスクの2種類のタスクを1つのプロセッサ内で扱い、前者のタスクの必要性能を確保し、後者のタスクを優先度に従って選択実行するよう構成されている。

【0025】

同図のようにプログラム実行装置は、タスクスケジューリング部10、プログラム記憶部20、タイマ制御部30及び実行制御部40より構成される。

タスクスケジューリング部10は、2種類のタスクのスケジューリングを行う。すなわち、前者のタスクに対して、タスクスケジューリング部10は、割当時間の指定を有するタスクのそれぞれに1つのタイムスロットを割り当て、予め定められた周期内において各タスクを少なくとも1回選択する時分割スケジューリングを行う。後者のタスクに対して、タスクスケジューリング部10は、一周期内における上記のタイムスロットの合計時間の残り時間において、優先度を有する複数タスクのうち1つのタスクを選択するスケジューリングを行う。ここで、タイムスロットとは、プロセッサにおけるプログラムを実行する時間を周期内において割当時間ごとに区切ったものである。

【0026】

プログラム記憶部20は、タスクスケジューリング部10によるスケジューリング対象となるタスクの本体であるプログラムとプログラムに関する情報とを記憶する。

【0027】

タイマ制御部30は、タスクスケジューリング部10から割当時間を設定される毎に、時間のカウントを開始し当該割当時間に達したときにタイムアウトを出力する。このタイムアウトは、タイムスロットの切り換えタイミングを知らせるために、タスクスケジューリング部10に通知される。

【0028】

実行制御部40は、タスクスケジューリング部10に選択されたタスクを実行する。実行制御部40はプロセッサのタスクを実行するハードウェアに相当する。

【0029】

〔タスクスケジューリング部10の構成〕

タスクスケジューリング部10は、タスク受付部11、タイムスロット記憶部12、タスク記憶部13、タイムスロット切換部14、タスク選択部15、mutex記憶部16及びmutex制御部17から構成される。

【0030】

<1. タスク受付部11>

タスク受付部11は、ユーザ操作やユーザプログラム等からの指示に従ってタスクの追加要求を受け付け、プログラム記憶部20から当該タスクに関する情報として、「タスク情報」、「優先度」及び「割当時間」を読み出して、当該タスクがスケジューリングの対象となるようにタイムスロット情報やタスク管理ブロックを生成し、タイムスロット記憶部12、タスク記憶部13に設定する。

【0031】

ここで、「タスク情報」は、プログラム開始アドレスとスタックポインタからなる。プログラム開始アドレスは、タスクが書き込まれた先頭のアドレスである。スタックポインタは、タスクの切換が生じた際にタスクの状態を一時的に退避する格納場所を示す位置情報である。

【0032】

「優先度」は、割当時間の指定を有するタスクか優先度を有するタスクかを区別する基準であり、且つ後者の場合の優先度を示すという2つの意味を持つ。

すなわち、優先度0のタスクは割当時間を有するタスク（以下種別Aのタスクと呼ぶ）、優先度が0でないタスクは優先度を有するタスク（以下種別Bのタスクと呼ぶ）であり、以下の意味を持つ。

【0033】

種別Aのタスクは、割当時間の指定を有するタスクであり、タスクスケジューリング部10において、タイムスロットに1対1で対応して割り当てられ、当該タイムスロットの割当時間の間は確実に実行される。実行時間の長さで処理性能が定まるので、いわゆるタイムドリブン型と呼ばれるタスクを種別Aとすべきである。例えば、動画データのデコード／エンコード処理、音声データのデコード

／エンコード処理など継続して一定の処理性能を必要とするタスクを種別 A とすることが望ましい。

【0034】

種別 B のタスクは、他の種別 B のタスクと共に 1 つのタイムスロットに割り当てられ、当該タイムスロットが保持する割当時間の間、優先度の高いタスクが実行される。ここでは、優先度は、高い方から順に 1、2、3、・・・という値をとるものとする。いわゆるイベントドリブン型と呼ばれるタスクを種別 B とすべきである。例えば、ユーザ操作をイベントとして文字や静止画からなるメニュー表示を行う処理など継続して一定の処理性能を必要としないが不定期に随時発生するタスクを種別 B とすることが望ましい。この場合、優先度は例えば厳しい応答速度を要求されるイベントを処理するタスクほど高くすればよい。

【0035】

「割当時間」は、タスクが種別 A のときにのみ有効で、タスクに対応するタイムスロットの割当時間を指定する値である。この割当時間と優先度とは、プログラムによって指定されたものとする。

【0036】

また、タスク受付部 11 は、ユーザ操作やユーザプログラム等からの指示があった場合や、タスクの実行が終了した場合に、タイムスロット記憶部 12、タスク記憶部 13 に記憶されている当該タスクのタイムスロット情報やタスク管理ブロックを削除する。

【0037】

< 2. タイムスロット記憶部 12 >

タイムスロット記憶部 12 は、タスクの切り換えの基準となるタイムスロットを生成するためのタイムスロット情報を記憶する。

【0038】

図 2 は、タイムスロット記憶部 12 内のタイムスロット情報と、タスク記憶部 13 内のタスク管理ブロックの具体例を示す図である。同図のように、タイムスロット記憶部 12 は、複数のタイムスロット情報 100、110、120、・・・を記憶する。タイムスロット情報 100 は、種別 B の複数のタスクが割り当て

られたタイムスロットを示す。このタイムスロット情報 100 は、種別 B のタスクが存在しなくても予め生成しておいてもよい。これ以外のタイムスロット情報 110、120、・・・は、種別 A のタスクが割り当てられたタイムスロットを示す。

【0039】

タイムスロット情報 100 は、タイムスロットの 1 つに対応し、割当時間 100a、フラグ 100b 及びポインタ 100c からなる。他のタイムスロット情報も同様である。

【0040】

割当時間は、当該タイムスロットに対応するタスクが実行することができる時間を示す。タスクが実際に実行された時間が割当時間に達すると次のタイムスロットに切り換えられる。種別 A のタスクに対応する割当時間 110a、120a は、タイムスロット情報の生成と同時に設定される。種別 B の複数タスクに対応する割当時間 100a は、周期レジスタ 18 に保持された一周期の時間から、他のタイムスロットの割当時間 110a、120a・・・の合計時間を引いた残り時間としている。この残り時間はタスクの追加、削除がある毎に変化する。

【0041】

実行フラグは、当該タイムスロットが有効か無効かを示す。タイムスロット情報の生成時には有効と設定され、タスク実行中にアクセス先の資源がロック状態にあって待ち状態になった時に無効と設定され、その後待ち状態から実行可能状態に戻った時に有効と設定される。実行フラグが無効を示す場合は、タイムスロット切換部 14 によってタイムスロット情報が存在しないものとみなされる。

【0042】

ポインタは、当該タイムスロットに対応するタスク管理ブロックを示している。

これらのタイムスロット情報 100、110、120・・・は配列を構成し、その順序はタイムスロットの生成順を表すものとする。

【0043】

< 3. タスク記憶部 13 >

タスク記憶部 13 は、タイムスロットに割り当てられたタスクに対応するタスク管理ブロック 200、201、・・・を記憶する。タスク管理ブロック 200、201、・・・は、それぞれ 1 つのタスクに対応し、当該タスクを管理するための情報である。

【0044】

タスク管理ブロック 200 は、タスク情報 200 a、優先度 200 b、リンクアドレス 200 c からなる。

タスク情報 200 a は、プログラム開始アドレス（又はプログラムを再開すべきアドレス）及びスタックポインタを含む。タスク管理ブロックの生成時には、タスク受付部によって入力された前記タスク情報を保持する。タスク切替時には、中断されたタスクの実行アドレス及び当該時点でのスタックポインタの値を示す。

【0045】

優先度 200 b は、当該タスクの優先度を示す。

リンクアドレス 200 c は、当該タスクが要素として実行キュー又は待ちキューに接続された場合に、実行キュー又は待ちキューにおける次の要素であるタスク管理ブロックへのポインタを保持する。

【0046】

具体的には、種別 A のタスクに対応するタスク管理ブロックでは、リンクアドレス（210 c 等）は、当該タスクが実行可能状態である場合は null（無効）であり、当該タスクが待ち状態である場合は待ちキューの次の要素を指すポインタである。種別 B のタスクに対応するタスク管理ブロックでは、リンクアドレス 200 c は、当該タスクが実行可能状態である場合は実行キューを形成する次の要素を指すポインタであり、当該タスクが待ち状態である場合は待ちキューを形成する次の要素を指すポインタである。

【0047】

< 4. タイムスロット切替部 14 >

タイムスロット切替部 14 は、現在のタイムスロットにおいてタスクの実行時間が割当時間に達したときに、タイムスロットの切替を行う。割当時間に達した

かどうかはタイマ制御部 30 によるタイムアウトによって通知される。通知を受けたタイムスロット切換部 14 は、次のタイムスロット情報を選択する。本実施例では、次に選択されるタイムスロット情報は、その配列順による。タイムスロット切換部 14 は、選択したタイムスロット情報から割当時間を取得してタイマ制御部 30 に設定する。これにより次のタイムスロットの割当時間のカウントが開始される。

【0048】

< 5. タスク選択部 15 >

タスク選択部 15 は、前記タイムスロット切換部 14 のタイムスロット切り換え時に、及び実行中のプログラムからの指示があった時に、現在実行中のタスクの実行アドレス、スタックポインタ等を当該タスクのタスク管理ブロックにタスク情報として退避し、次に実行すべきタスクのタスク管理ブロックからタスク情報を取り出して、実行制御部 40 へ出力する。これと同時に、実行中のタスクのコンテキスト（レジスタデータ等）の退避と、次に実行すべきタスクのコンテキストの復帰もなされる。これにより次に実行すべきタスクが実行状態になる。

【0049】

図 4 は、タイムスロット切換部 14 及びタスク選択部 15 によるタスク切換の様子を示す説明図である。同図において、 T は周期レジスタ 18 に保持された周期、 t_0 は種別 B のタスクに対応するタイムスロット情報 100 に対応するタイムスロットであり、その長さは割当時間を示す。 $t_1 \sim t_n$ はそれぞれ種別 A の n 個のタスクに対応する n 個のタイムスロット情報 110、120・・・に対応するタイムスロットであり、その長さは割当時間を示す。 S は、タイムスロット切換部 14 及びタスク選択部 15 によってタスク切換を行うスケジューリング処理を示す。同図のように、タイムスロット t_0 を開始時のスケジューリング処理では、種別 B の複数のタスク B1、B2、B3 のうち優先度の最も高いタスク B1 が選択され実行される。タイムスロット t_1 の開始時のスケジューリング処理では、タイムスロット情報 110 に対応するタスク A1 が実行される。タイムスロット $t_2 \sim t_n$ についても同様である。このように種別 A のタスクは 1 つの周期 T の間に 1 回は必ず実行される。

【0050】

また、図5は、タイムスロット t0において実行中のタスクが終了した場合のタスク切替の様子を示す説明図である。同図においてタスク B1は、自身の処理を完了した場合、タスク選択部15にその旨を通知する。この通知によって、タスク B1の削除処理（図中のE）と、タスクの切替処理（同S1）とがなされる。削除処理Eでは、タスク B1が終了したので、タスク B1のタスク管理ブロックが削除される。切替処理S1では、タスク B1のタスク管理ブロック削除後に最も優先度の高いタスク B2がタスク選択部15によって選択され、実行される。このように、種別Bの複数のタスクが割り当てられたタイムスロット t0では、優先度の高いタスクから順に実行される。

【0051】

< 6. mutex記憶部16 >

mutex (mutual exclusion: 相互排除) は、2つ以上のタスクが1つの資源を競合してアクセスする場合に調停を行う機構をいい、mutex制御部17に含まれる機能であり、資源のロック状態及びロック解除状態を管理するオブジェクト（プログラム）によって実現される。この機構は、2つ以上のタスクからのアクセスが競合する可能性のある資源毎に設けられる。あるタスクがmutexのロック操作に成功すると、mutexは（又は資源は）ロック状態になり他のタスクがロック操作を行ってもロック解除状態になるまで待たされることになる。これによりロック操作に成功したタスクは対応する資源を専有することができる。ロック操作を行ったタスクがロック解除操作を行うと、他のタスクがロック操作を行うことができるようになる。実行中のタスクは、資源をアクセスする場合、当該資源に対応するmutexをロック操作に成功した場合のみ資源をアクセスすることができる。既にロック状態になっている場合には、ロック解除状態になるまで待たされることになる。

【0052】

図3は、mutex記憶部16に記憶されるmutex管理ブロック300、301・・・と、タスク記憶部13に記憶される待ちキューとを示す図である。

同図においてmutex管理ブロック300、301・・・は、2つ以上のタスク

からのアクセスが競合する可能性のある個々の資源に対応して設けられる。

【0 0 5 3】

mutex管理ブロック 3 0 0 はロック変数 3 0 0 a 及びポインタ 3 0 0 b を含む。

ロック変数 3 0 0 a は、2 つの状態（1 : ロック状態、0 : ロック解除状態）の何れかを示す。初期値は 0 であり、ロック解除状態からロック状態にするにはタスクがロック操作を行う必要がある。ロック状態からロック解除状態にするには、ロック操作を行ったタスクのみが操作を許される。

【0 0 5 4】

ポインタ 3 0 0 b は、ロック状態においてロック操作を行ったタスク（ロック操作に失敗したタスク）を示す待ちキューの先頭のタスク管理ブロックを指す。待ちキューに接続されたタスク管理ブロック 2 0 0'、2 0 1'・・・は、ロック操作に失敗したタスクに対応し、それぞれリンクアドレス 2 0 0' c、2 0 1' c・・・によって順に接続された待ちキューを形成する。

【0 0 5 5】

< 7. mutex制御部 1 7 >

mutex制御部 1 7 は、mutexのロック操作及びロック解除操作が行われるときに、ロック変数の操作を行うとともに、タスク選択部 1 5 へのタスク管理ブロックの操作要求やタイムスロット切換部 1 4 への通知及び実行フラグの操作要求を行う。

【0 0 5 6】

図 6 は、種別 B のタスクがロック操作に失敗した場合の様子を示す説明図である。同図では、タイムスロット t 0 において実行中のタスク B 1 がアクセスしようとする資源に対応するmutexに対してロック操作（図中の L : ロック処理）を行って、失敗した場合を示している。当該mutexが既にロック状態である場合にタスク B 1 はロック操作に失敗することから、同図のようにロック処理 L によってタスク記憶部 1 3 の実行キューから待ちキューに移動させられ、さらに、タスク切換処理 S 1 により、タスク B 2 に切り換えられる。タスク B 1 は、当該mutexがロック状態からロック解除状態になるまで、実行キューから除外されるので

、スケジューリングの対象から除外される。

【0 0 5 7】

このように種別Bのタスクに対応するタイムスロット t 0 では、実行中のタスクが待ち状態になった場合に、次順位のタスクに切り換えられる。

また、タスク B 1 がロック処理においてロック操作に成功した場合は、タスク B 2 に切り換えられることなく、タイムスロット t 0 の間、実行を続ける（資源にアクセスする）ことになる。

【0 0 5 8】

図 7 は、種別 A のタスクがロック操作に失敗した場合の様子を示す説明図である。同図では種別 A のタスク A 2 がロック操作に失敗して待ちキューに移動させられ場合、タイムスロット t 2 の割当時間に達していなくても、スケジューリング処理（同図 S）によりタイムスロットが切り換えられる。種別 A のタスクに対応するタイムスロットにはタスクが 1 つしか割り当てられていないから、タイムスロットの残り時間があってもタイムスロットを切り換えている。

【0 0 5 9】

また、タスク A 2 がロック操作に成功した場合には、タイムスロット t 2 の割当時間に達した時に次のタイムスロットに切り換えられる。

図 8 は、タスクがロック解除する場合の様子を示す説明図である。同図に示すように実行中のタスク A 1 がロック状態にあるmutexをロック解除した場合、ロック解除処理 R によって当該mutexに対応する待ちキュー R のタスク B 1 は実行キュー B に戻される。これによりタスク B 1 は、それ以後に対応するタイムスロット t 0 が選択された場合に実行されることになる。

【0 0 6 0】

図 9 は、タスクがロック解除する場合の様子を示す他の説明図である。同図では実行中のタスク A 2 がロック状態にあるmutexをロック解除した場合、ロック解除処理 R によって当該mutexに対応する待ちキュー S のタスク B 2 は実行キュー B に戻される。図 8 とは異なり、タスク B 2 は、タスク B 1 と B 3 の間に戻されている。これは、優先度の順に並ぶように実行キューに戻しているからである。このケースは、タスク B 2 が待ちキューにある間に、それよりも優先度の高い

タスク B 1 が生成された場合などにあたる。これ以降にタイムスロット t 0 が選択された場合、優先度の最も高いタスク B 1 が選択されて実行されることになる。

【0061】

〔処理の詳細〕

以下、本発明のプログラム実行装置における処理の詳細として、スケジューリング処理（図4～図9中のS）、mutexロック処理（図6、図7中のL）及びmutexロック解除処理（図8、9中のR）、タスク切換処理（図5、図6中のS1）、タスク追加処理、タスク終了処理（図5中のE）について詳細に説明する。

【0062】

<1. スケジューリング処理>

図10は、スケジューリング処理の詳細を示すフローチャートである。同図のスケジューリング処理は、タイマ制御部30がタイムアウトをタイムスロット切換部14に通知したとき、及び、実行中のタスクがロック操作に失敗したとき（図7参照）に呼び出されて開始する。図中、iは実行フラグが無効になっているタイムスロット情報をカウントするための変数、(n+1)は存在するタイムスロット情報の数を表す。

【0063】

まず、タイムスロット切換部14は、カウンタとして用いる変数iを0に初期化し（S401）、現在実行しているタスクの状態をタスク管理ブロックに退避する（S402）。

【0064】

次に、タスク選択部15は、変数iに1を加算し（S403）、タイムスロット情報の配列における次順のタイムスロット情報を選択し、次の要素がない場合は、タイムスロット情報配列の先頭の要素を選択し、選択したタイムスロット情報を取得する（S404）。さらに、タイムスロット切換部14は、選択したタイムスロット情報に含まれる割当時間を取り出し、タイマ制御部30に設定する（S405）。これにより、タイマ制御部30はカウントを開始し、割当時間に達しタイムアウトするまでカウントする。

【0065】

さらに、タイムスロット切換部14は、取得したタイムスロット情報がタイムスロット情報100であるか否かを判定する。つまり次に実行すべきタスクが種別Bであるか種別Aであるかを判定する（S406）。

【0066】

判定の結果が種別Aである場合には、当該タイムスロット情報中のポインタが指すタスク管理ブロックを取得し（S407）、タイムスロット情報中の実行フラグが1（有効）であれば（S408）、タスク管理ブロック中のタスク情報に従って、当該タスクの状態を復帰させる（S411）。これにより種別Aのタスクに切り換えられる。

【0067】

また、判定の結果が種別Bである場合には、タイムスロット切換部14は、タイムスロット情報100中のポインタにタスク管理ブロックの実行キューが接続されているか否か（ポインタが有効か無効か）を判定する（S412）。実行キューが接続されていると判定された場合には、優先度が最も高い先頭のタスク管理ブロックを取得し（S413）、当該タスク管理ブロック中のタスク情報に従って、当該タスクの状態を復帰させる（S411）。これにより種別Bのタスクのうち最も高い優先度のタスクに切り換えられる。また、実行キューが接続されていないと判定された場合は、実行可能な種別Bのタスクが存在しない。この場合、次のタイムスロット情報を選択するためにS403へ戻る。

【0068】

上記S408において実行フラグが無効であれば（S408：no）、タイムスロット切換部14は再度S403、S404に戻って（S409：no）次のタイムスロットを選択する。さらに、この繰り返しによりn+1個のタスク管理情報が無効であった場合（S409：yes）、つまり、実行可能なタスクが1つも存在しない場合には、プログラム実行装置を省電力状態に移行させる（S410）。

【0069】

このように、タスクスケジューリング部10によるスケジューリング処理では、種別Aのタスクと種別Bのタスクとでは異なるスケジューリングをしている。

すなわち、タスクスケジューリング部10は、種別Aのタイムスロット（タイムスロット情報1、2、・・・に対応するタイムスロット）が選択された場合、当該タイムスロットに1つだけ割り当てられた種別Aのタスクに切り換える。また、タスクスケジューリング部10は、種別Bのタスクが割り当てられているタイムスロット（タイムスロット情報100に対応するタイムスロット）が選択された場合、種別Bの複数のタスクのうち優先度の最も高いタスクに切り換える。

【0070】

また、図5及び図6に示した切換処理S1は、図10のスケジューリング処理とほぼ同様とすればよい。例えば、切換処理S1では図10スケジューリング処理におけるS412から開始するようにしてもよい。

【0071】

<2. タスク追加処理>

図11は、タスクスケジューリング部10におけるタスク追加処理の詳細を示すフローチャートである。このタスク追加処理はユーザ操作やユーザプログラム等からのタスク追加要求が発生した時点で開始される。

【0072】

タスク追加の要求を受けて、タスク受付部11は、追加すべきタスクのタスク情報、優先度及び割当時間を取得し（S501）、追加すべきタスクの種別が種別Aであるか種別Bであるか（当該優先度が0であるか0でないか）を判定する（S502）。

【0073】

追加すべきタスクが種別Aであると判定された場合、タスク受付部11は、タイムスロットの追加が可能かどうかを検査する。すなわち、タスク受付部11は、タイムスロット記憶部12から種別Aのタスクに対応するタイムスロット情報110、120、・・・、内の割当時間を読み出して、その合計を算出し、当該合計に追加すべきタスクの割当時間を加えた総和が、周期レジスタ18に保持された周期値を超えていなければ、当該タスクに対応するタイムスロットの追加が可能であると判定する（S503）。当該周期値を超えている場合は、追加要求に対するエラーを返す（S504）。

【 0 0 7 4 】

追加可能と判定されたならば、タスク受付部 1 1 は、割当時間と実行フラグ（この時点では 0）とを設定したタイムスロット情報を生成して（S 5 0 5）、タスク情報と優先度とを含むタスク管理ブロックを生成する（S 5 0 6）。

【 0 0 7 5 】

タイムスロット情報とタスク管理ブロックが生成した後、タスク受付部 1 1 は、その対応関係を示すためにタスク管理ブロックの格納位置をタイムスロット情報のポインタにセットする（S 5 0 7）。最後に当該タイムスロットの実行フラグを有効にする（S 5 0 8）。

【 0 0 7 6 】

上記 S 5 0 2 において追加すべきタスクが種別 B であると判定された場合、タスク受付部 1 1 は、追加すべきタスクのタスク管理ブロックを生成し（S 5 0 9）、タイムスロット情報 1 0 0 に対応する実行キューに追加する。実行キューへの追加は、実行キューの先頭から優先度の高い順に並ぶように、追加すべき位置を探索して当該タスク管理ブロックを追加する（S 5 1 0）。また、タイムスロット情報 1 0 0 内の実行フラグに 1 をセットする（S 5 0 8）。

【 0 0 7 7 】

このように、追加処理では、既に存在するタイムスロット情報及びタスク管理ブロック（図 2 参照）に対して、追加すべきタスクが種別 A である場合は新たなタイムスロット情報と新たなタスク管理ブロックを追加し、追加すべきタスクが種別 B である場合は実行キューの優先度に応じた位置に新たなタスク管理ブロックを追加する。

【 0 0 7 8 】

また、図 5 に示したタスク削除処理 E は、削除すべきタスクが種別 A である場合は、当該タスクに対応するタイムスロット情報とタスク管理ブロックを削除し、削除すべきタスクが種別 B である場合は、当該タスクに対応するタスク管理ブロック実行キューから削除する。

【 0 0 7 9 】

< 3. mutex ロック処理 >

図 1 2 は、mutex制御部 1 7 によるロック処理の詳細を示すフローチャートである。このロック処理は、図 6 及び図 7 のロック処理 L であり、実行中のタスクが資源の 1 つをアクセスする直前に当該資源に対応するmutexをロックするために、実行中のタスクによって呼び出される処理である。

【 0 0 8 0 】

実行中のタスクに呼び出されるとmutex制御部 1 7 は、資源に対応するmutex管理ブロックに対してロック変数の読み出しと有効値” 1 ”の書き込みを行う（S 6 0 1）。この読み出しと書き込みとの間に、他のタスクが同じ処理を行うことによって値に矛盾が生じることを防ぐために、読み出しと書き込みを一命令で行っている。これは、プロセッサに実装されている read-modify-write 命令により実現可能である。

【 0 0 8 1 】

読み出されたロック変数が 0（ロック解除状態）ならば（S 6 0 2 : yes）、ロック操作の成功であり、上記書き込みにより” 1 ”をセット済であるのでロック処理を終了する。これにより、ロック処理を呼び出したタスクに実行制御が復帰し、当該タスクはmutexに対応する資源を専有してアクセスすることが可能になる。

【 0 0 8 2 】

読み出されたロック変数が 1（ロック状態）ならば（S 6 0 2 : no）、すでに他のタスクにより資源が専有されているため、呼び出し元のタスクはロック解除状態になるまで、当該資源をアクセスすることができない。この場合、mutex制御部 1 7 は、呼び出し元のタスクが種別 A であるか種別 B であるかを判定する。

【 0 0 8 3 】

判定結果が種別 A（優先度が 0）ならば（S 6 0 3 : yes）、対応するタイムスロット情報の実行フラグを無効にする（S 6 0 4）。これにより当該タイムスロット情報はスケジューリング対象から除外される。加えて、mutex制御部 1 7 は呼び出し元のタスクのタスク管理ブロックを実行キューから外して、当該mutex管理ブロックの待ちキューにつなぐ（S 6 0 5）。待ちキューにつなぐのは、どのmutexのロック解除待ち（どの資源の専有状態解除待ち）になっているかを

示すためである。さらに、mutex制御部17は、スケジューリング処理Sを呼び出す（S606）。これにより当該タイムスロットは割当時間の途中でも強制的に次のタイムスロットに切り換わる（図7参照）。

【0084】

なお、S604において実行フラグを無効にする代わりに、タイムスロット情報のポインタが、種別Bのタスクの実行キューの先頭を指すように変更してもよい。これにより、種別Bのタスクが当該タイムスロットでも実行されることになる。

【0085】

判定結果が種別B（優先度が0でない）ならば（S603：no）、mutex制御部17は、呼び出し元のタスクのタスク管理ブロックをは実行キューから外す（S607）。これにより当該タスクはスケジューリング対象から除外される。さらに、どの資源の専有状態が解除されるのを待っていることを示すために、前記タスク管理ブロックを mutex 管理ブロックの待ちキューに追加する（S608）。さらに、タスクの切替処理S1を呼び出すことにより次順位のタスクに切り換える（S609）。これによりタイムスロットの割当時間の残り時間において次順位のタスクが実行される（図6参照）。

【0086】

< 4. mutexロック解除処理 >

図13は、mutex制御部17によるmutexロック解除処理の詳細を示すフローチャートである。この処理は、図8及び図9に示したロック解除処理Rであり、実行中のタスクが1つ資源のアクセスを終えたときに、当該実行中のタスクによって呼び出される処理である。

【0087】

呼び出されると、mutex制御部17は、当該資源に対応するmutex管理ブロックの待ちキューから先頭のタスク管理ブロックを取り出し（S701）、取り出したタスク管理ブロック内の優先度から待ち状態にあったタスクの種別を判定する（S702）。mutex制御部17は、優先度が0（種別A）ならば、取り出したタスク管理ブロックに対応するタイムスロット情報中の実行フラグを1（有効）

にする（S703）。取り出したタスク管理ブロックに対応するタイムスロット情報は、例えば、実行フラグが0になっているタイムスロット情報に接続されているタスク管理ブロックのうち、取り出したタスク管理情報と同じタスク情報をもつタスク管理ブロックを検索することにより特定すればよい。さらに、mutex制御部17は、当該mutex管理ブロック内のロック変数に0を書き込む（ロック解除状態にする）（S705）。

【0088】

また、待ちキューから取り出したタスク管理ブロックの優先度が0でない（種別B）ならば、mutex制御部17は、当該タスク管理ブロックを種別Bのタスクの実行キューに追加する。実行キューへの追加は、先頭から優先度の高い順に並ぶ位置に追加し（S704）、当該mutex管理ブロック内のロック変数に0を書き込む。

【0089】

このように、待ち状態になっていたタスクは、実行可能状態に戻るので、スケジューリングの対象となり、次に実行状態になったとき再度資源のロック操作を行うことができる。

【0090】

以上説明してきたように本実施の形態におけるプログラム実行装置によれば、種別Aのタスクは1対1でタイムスロットを割り当てられる結果、一周期内で必ず選択実行されるので継続して処理性能を保証することができる。他方、種別Bのタスクは多対1で1つのタイムスロットを割り当てられる結果、継続して処理性能が保証されないが、優先度の高いものから実行されることになる。その結果、プログラマは、継続して処理性能を満たす必要のあるタスクに対しては優先度を考慮する必要もなく種別Aに分類するだけでよく、処理性能を要求しないタスクに対しては種別Bに分類すればよい。これにより、全てのタスクに対して個別に優先度を付与する必要がなくなり、処理性能を確保するためのプログラム設計を容易にすることができ、プログラム設計の柔軟性を確保することができる。

【0091】

なお、上記実施の形態では割当時間の指定はタスク情報に含まれている例を説

明したが、タスク受付部 11 が外部から割当時間の指定を受ける構成としてもよい。また、割当時間の指定ない場合は既定の割当時間としてもよい。

【0092】

また、種別 A のタスクには 1 つのタスクに 1 つのタイムロットが割り当てられているが、1 つのタスクに 2 つ以上のタイムスロットを割り当てる構成としてもよい。

【0093】

さらに、種別 B のタスクに対して、複数のタスクに 1 つのタイムロットが割り当てられているが、固定された 2 つ以上のタイムスロットの何れかに割り当てるようにしてもよい。例えば、優先度が奇数のタスクをタイムスロット A に、偶数のタスクをタイムスロット B に割り当てる構成としてもよい。

【0094】

また、mutex 管理ブロック内にロックしたタスク名を記録する所有タスク欄を設け、ロック解除されたとき、所有タスク欄を待ちキューから取り出したタスク名に書き換えて、ロック変数を有効なままにしてもよい。この場合、所有タスク欄に記憶されたタスクはロック操作を行わなくてもよく、確実に資源を使用することができるようになる。

【0095】

最後に、プログラム実行装置の変形例を図 14 に示す。図 1 に示したプログラム実行装置の構成と比較して、タイムスロット切替と同時に切替可能な 8 つのレジスタセット RS0 ～ RS7 を追加した点が異なっている。

【0096】

1 つのレジスタセットは 1 つのタイムスロットに対応させて、タスクのコンテキストを保持する。

種別 B の複数のタスクは 1 つのレジスタセットに対応するので、種別 A のタスクのタイムスロットに切り換える際に、タスクスケジューリング部 10 は、上記実施の形態と同様にレジスタセット RS0 に対する退避及び復帰を行う。

【0097】

種別 A のタスクは 1 つのレジスタセットに対応するので、種別 A のタスクのタ

タイムスロットに切り換える際に、タスクスケジューリング部10は、コンテキストの退避及び復帰の代わりにレジスタセットの切換を行う。この場合、コンテキストの退避及び復帰が不要となるので、タスク切換を高速化することができる。このことから、周期レジスタ18の周期及びタイムスロットの割当時間を極端に短くすることができる。例えば、種別Aのタイムスロット時間を退避及び復帰に要していた時間程度まで短くすることができ、一周期を短縮することができる。

【0098】

なお、図14におけるレジスタセットの数は任意の個数でよく、タイムスロットの数がレジスタセットの数よりも多い場合は、例えば、レジスタセットの何れかを複数のタイムスロットが兼用し、当該レジスタセットに対しては、タイムスロット切り換え時にコンテキストの退避及び復帰を行う構成とすればよい。

【0099】

また、図14においてレジスタセットの数を2としてもよい。その場合、レジスタセットの1つを実行中のタスクが使用する表レジスタセットとし、他の1つを次に実行されるタスクのコンテキストをバックグラウンド処理により復帰させておくための裏レジスタセットとしてもよい。この構成によれば、アイドル状態を利用してバックグラウンド処理によりコンテキストの復帰をすることができ、裏と表の2つのレジスタセットを効率よく切り換えることによる高速化を図ることができる。

【0100】

さらに、レジスタセットの数が2のとき、ロック解除によって待ち状態から実行可能状態に復帰したタスクが存在し 当該タスクが次に実行されるタスクである場合には、裏レジスタセットに復帰したコンテキストが次に実行されるタスクのコンテキストと食い違う可能性がある。これに対しては以下の構成とすればよい。すなわち、ロック解除処理（図13参照）においてS705の直前に、待ち状態から実行可能状態に復帰したタスクが存在することを知らせる通知フラグを当該タスクのタスク管理ブロック内に設定するステップを追加し、スケジューリング処理（図10参照）において、S411の直前に、通知フラグが待ち状態から復帰したタスクであることを示していれば、当該タスクのコンテキストを裏レ

ジスタセットに復帰してから、レジスタセットの裏と表とを切り換えるステップを追加した構成とすればよい。

【0101】

【発明の効果】

以上説明してきたように本発明のタスク切換装置によれば、プログラマは、継続して処理性能を満たす必要のあるタスクに対しては優先度を考慮する必要もなく第1タイプに分類するだけでよく、処理性能を要求しないタスクに対しては第2タイプに分類すればよい。これにより、全てのタスクに対して個別に優先度を付与する必要がなくなり、処理性能を確保するためのプログラム設計を容易にすることができ、プログラム設計の柔軟性を確保することができるという効果がある。

【0102】

また、第2タイプのタスクは優先度の高いものから実行されるので、プログラマは、第1タイプのタスクに対して優先度を考慮する必要がなく、継続した処理性能を要求しないタスクを第2タイプに分類して優先度を付与すればよく、プログラム設計及び変更を容易にすることができる。

【図面の簡単な説明】

【図1】

本発明の実施の形態におけるタスク切換を行うプログラム実行装置の主要部の構成を示すブロック図である。

【図2】

タイムスロット記憶部内のタイムスロット情報と、タスク記憶部内のタスク管理ブロックの具体例を示す図である。

【図3】

mutex記憶部に記憶されるmutex管理ブロックと、タスク記憶部に記憶される待ちキューとを示す図である。

【図4】

タイムスロット切換部及びタスク選択部によるタスク切換の様子を示す説明図である。

【図 5】

種別 B のタスクが終了した場合のタスク切換の様子を示す説明図である。

【図 6】

種別 B のタスクがロック操作に失敗した場合の様子を示す説明図である。

【図 7】

種別 A のタスクがロック操作に失敗した場合の様子を示す説明図である。

【図 8】

タスクがロック解除する場合の様子を示す説明図である。

【図 9】

タスクがロック解除する場合の様子を示す他の説明図である。

【図 10】

スケジューリング処理の詳細を示すフローチャートである。

【図 11】

タスク追加処理の詳細を示すフローチャートである。

【図 12】

mutex制御部によるロック処理の詳細を示すフローチャートである。

【図 13】

mutex制御部によるmutexロック解除処理の詳細を示すフローチャートである。

【図 14】

プログラム実行装置の変形例の構成を示す図である。

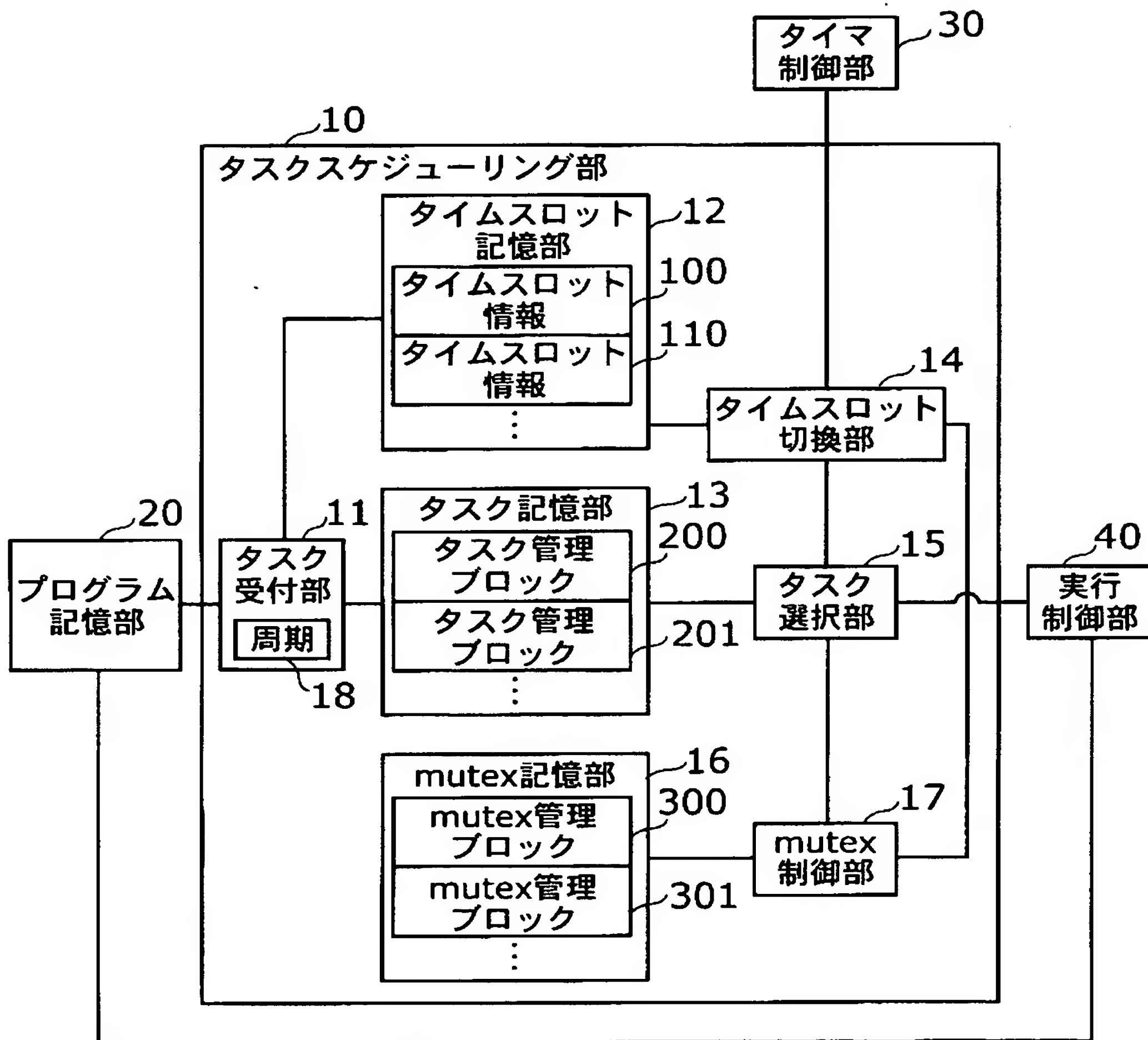
【符号の説明】

- 10 タスクスケジューリング部
- 11 タスク受付部
- 12 タイムスロット記憶部
- 13 タスク記憶部
- 14 タイムスロット切換部
- 15 タスク選択部
- 16 mutex記憶部
- 17 mutex制御部

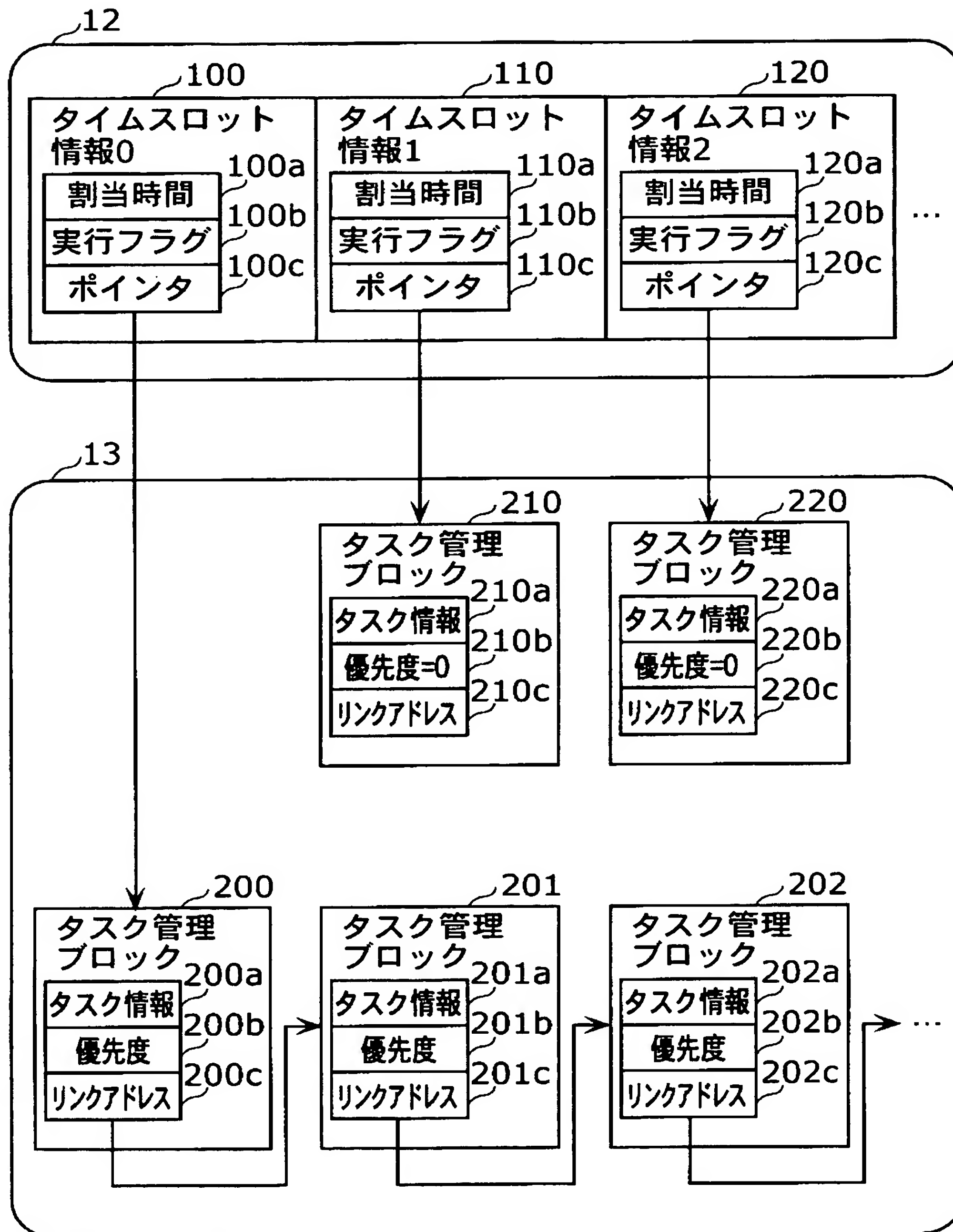
18 周期レジスタ
20 プログラム記憶部
30 タイマ制御部
40 実行制御部
100、110、120 タイムスロット情報
100a、110a、120a 割当時間
100b、110b、120b 実行フラグ
100c、110c、120c ポインタ
200、201、202、210、220 タスク管理ブロック
200a、201a、202a、210a、220a タスク情報
200b、201b、202b、210b、220b 優先度
200c、201c、202c、210c、220c リンクアドレス
300、301 mutex管理ブロック
300a、301a ロック変数
300b、301b ポインタ

【書類名】 図面

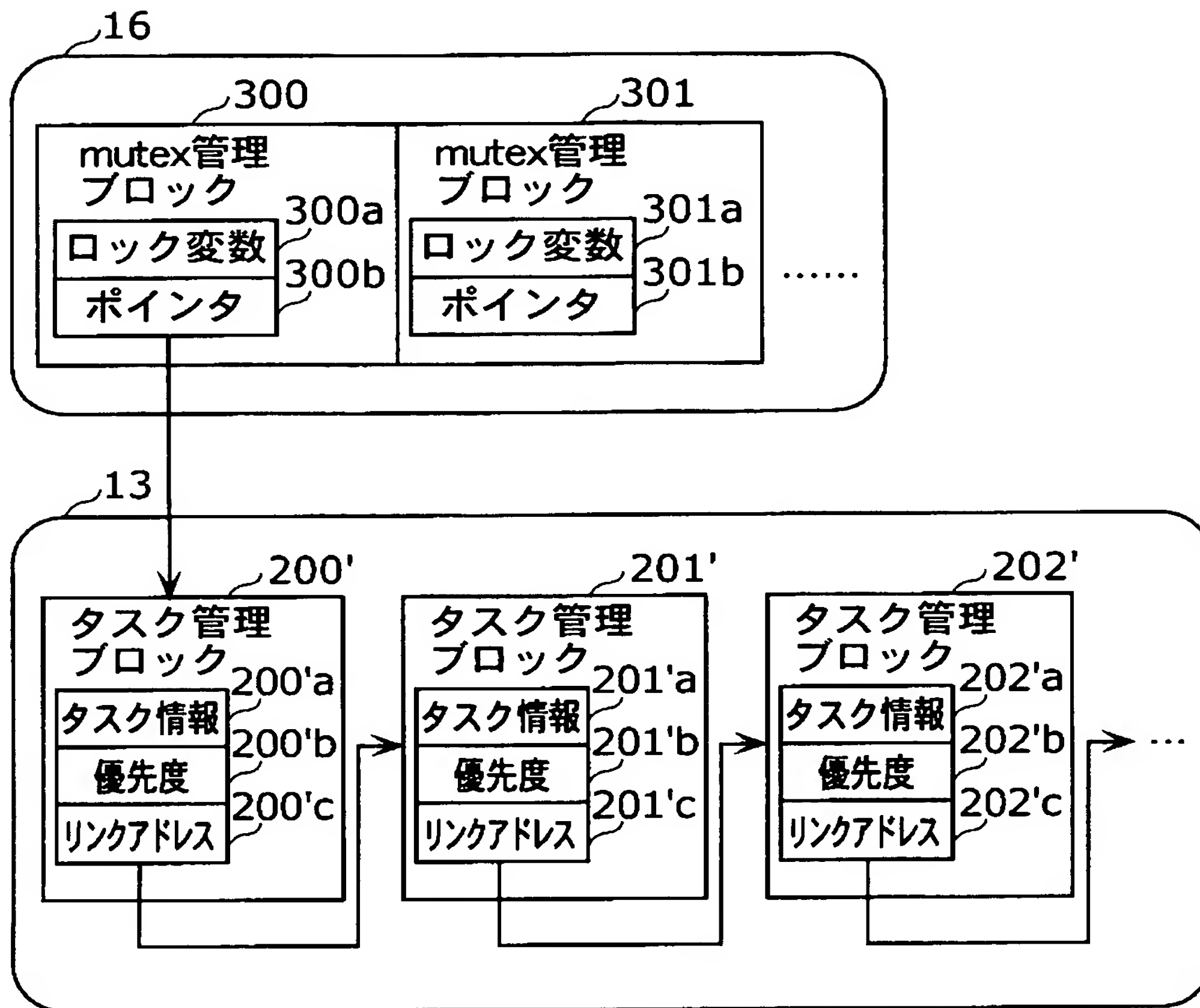
【図 1】



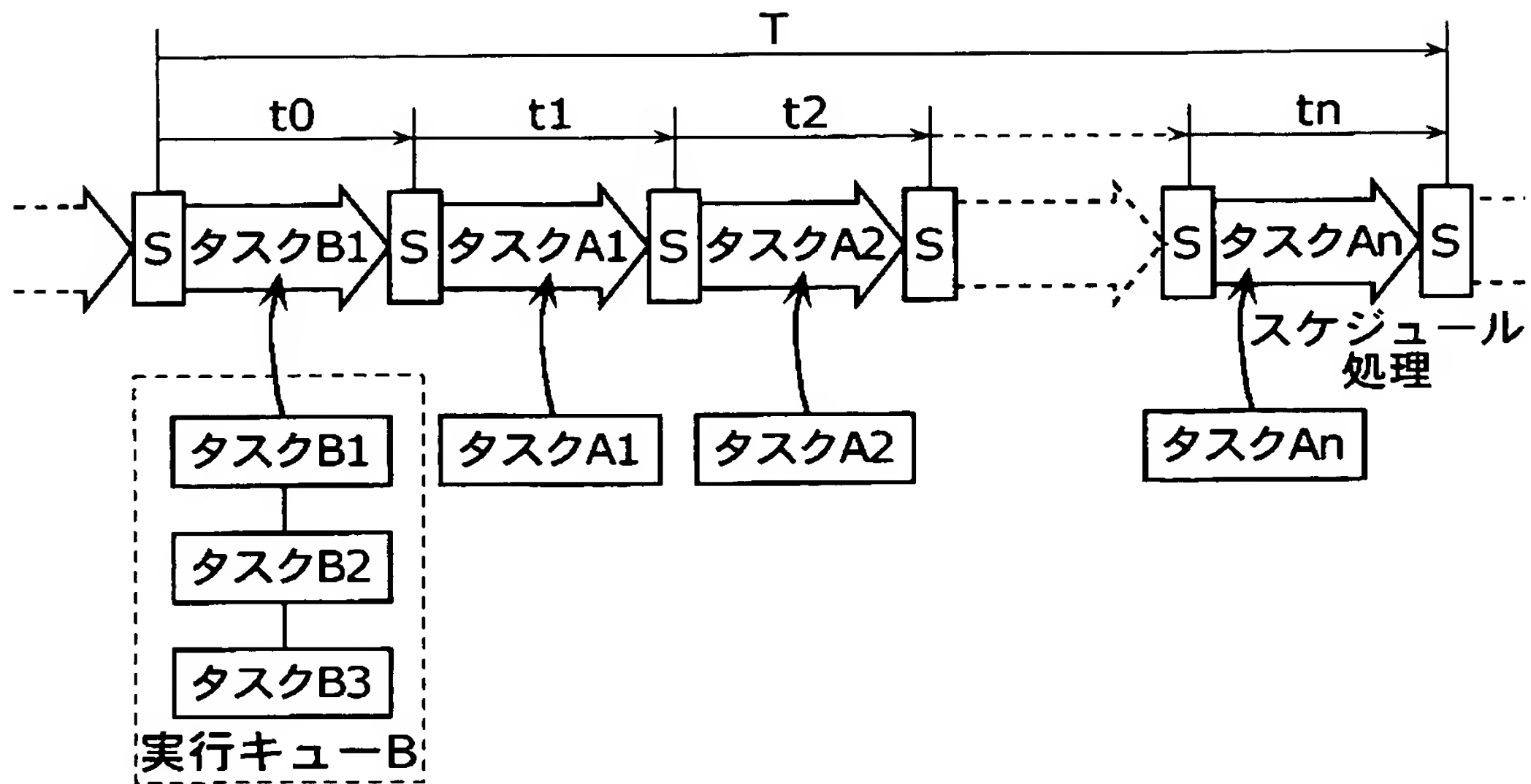
【図 2】



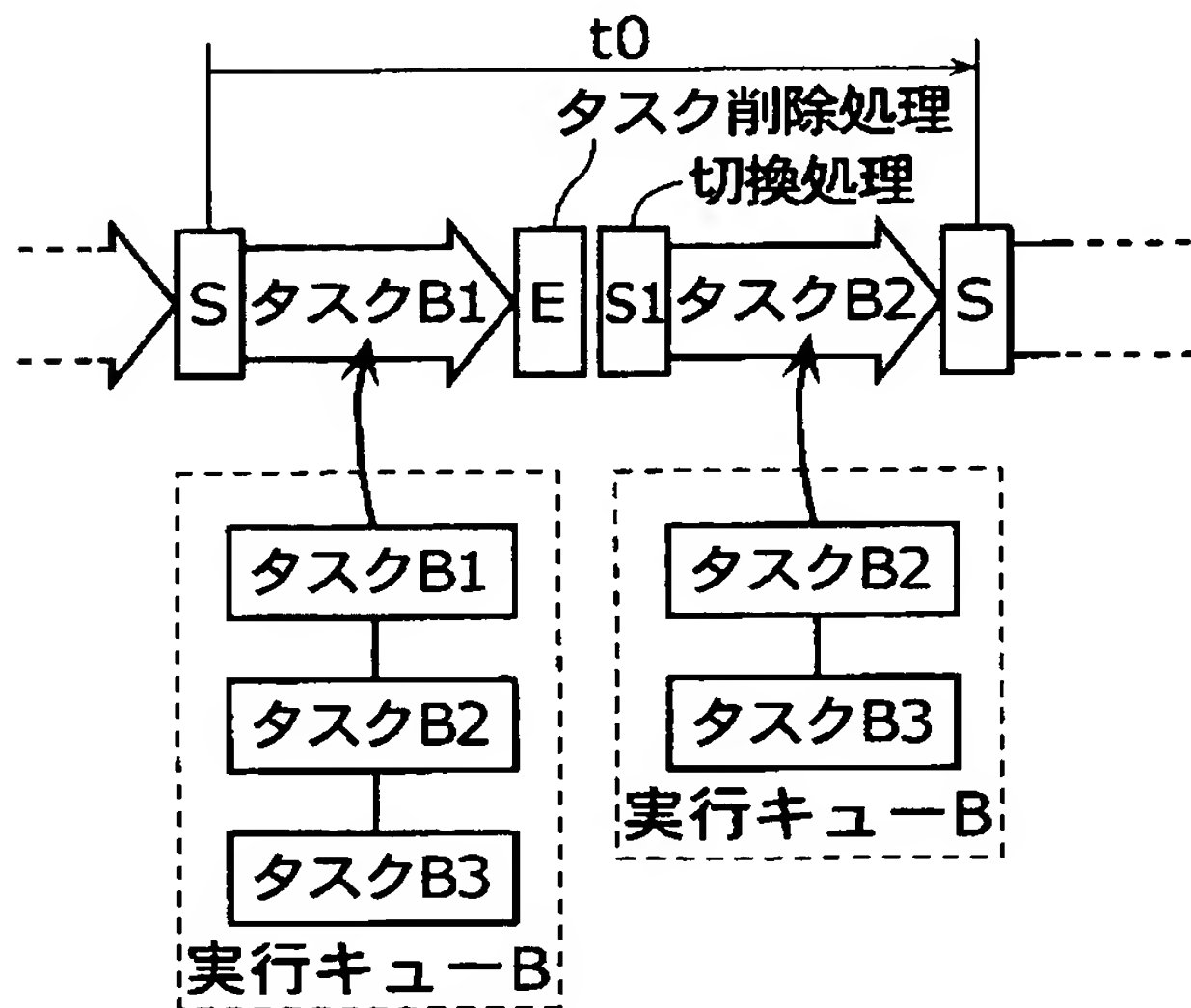
【図 3】



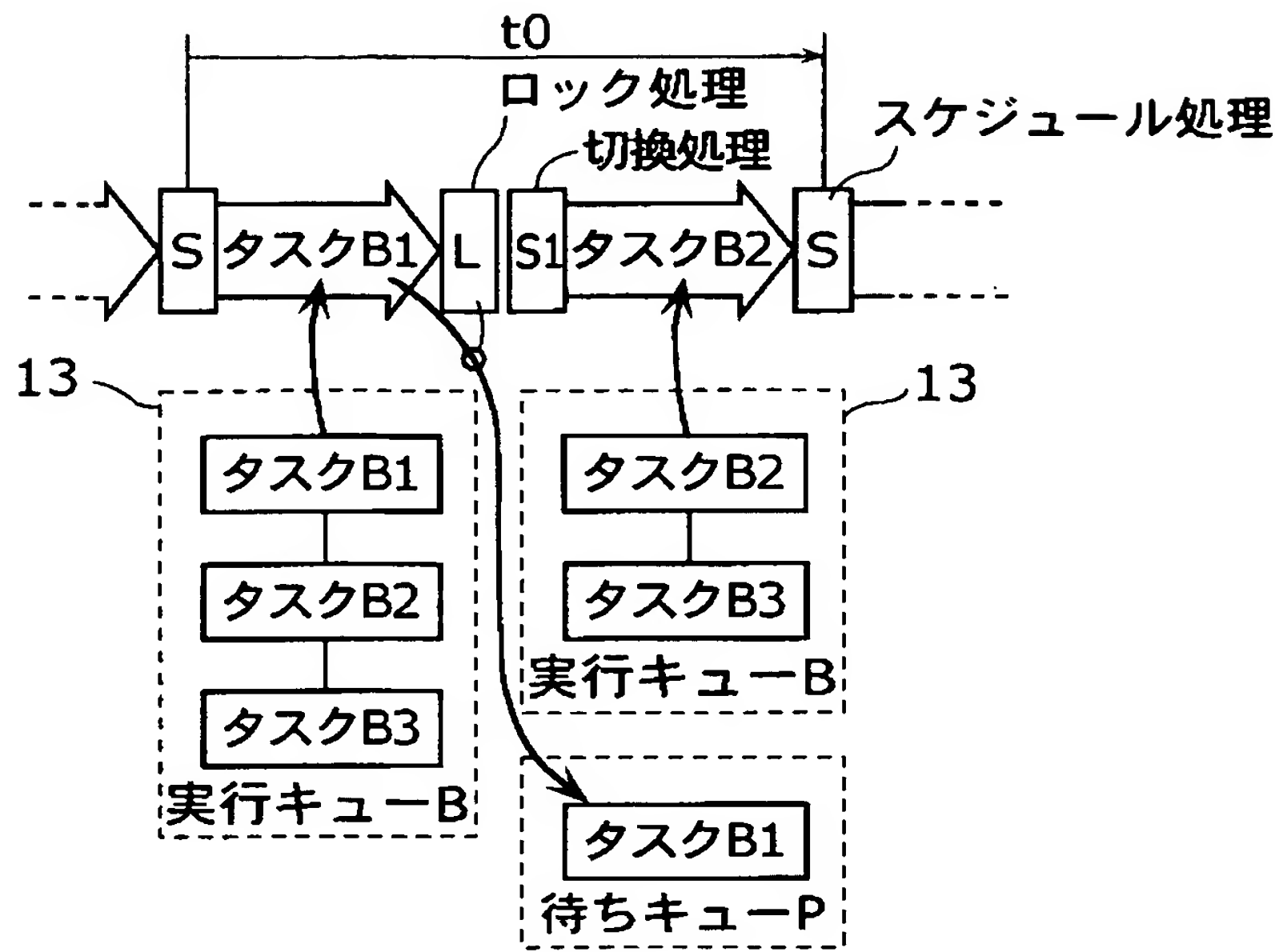
【図 4】



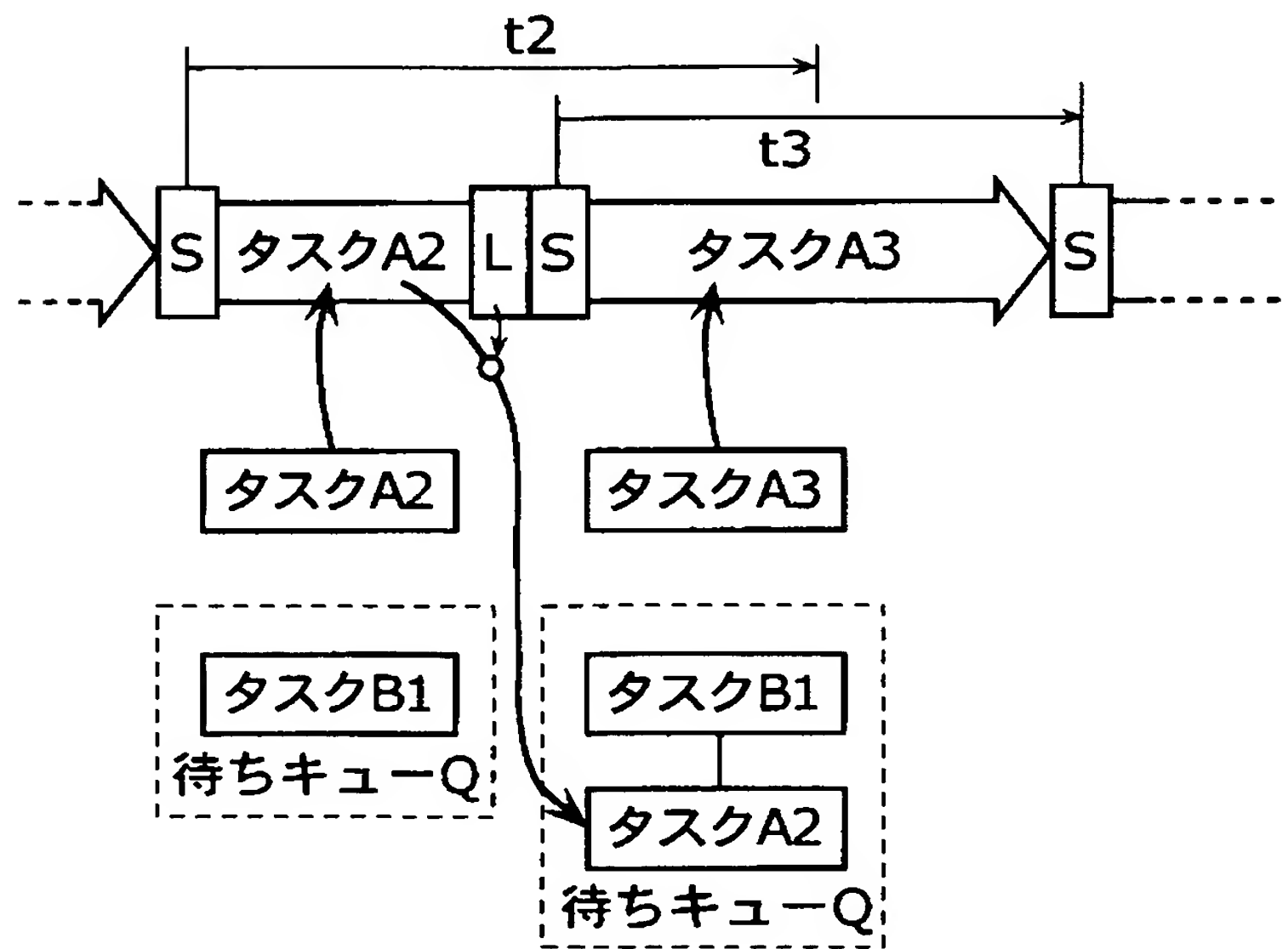
【図 5】



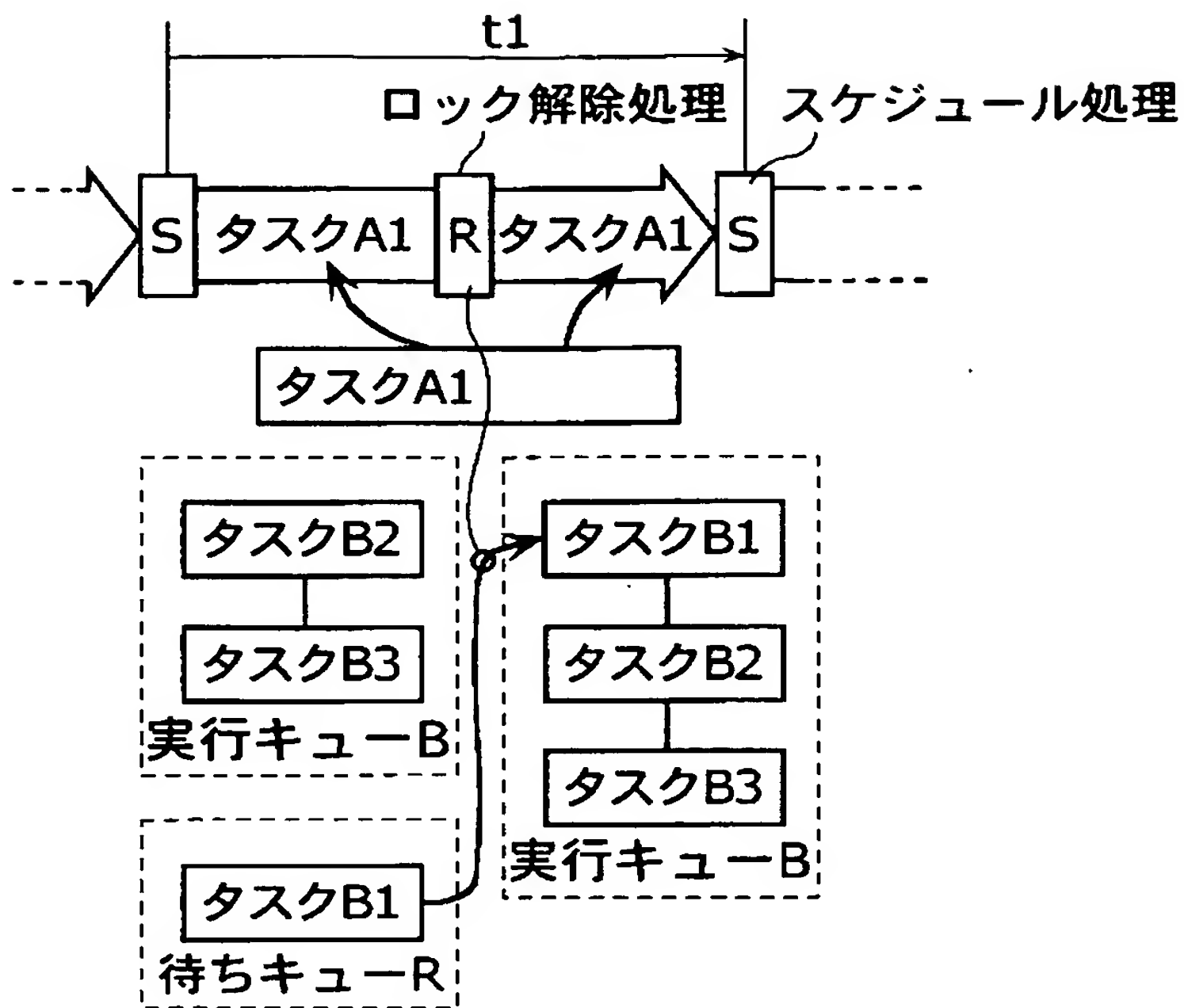
【図 6】



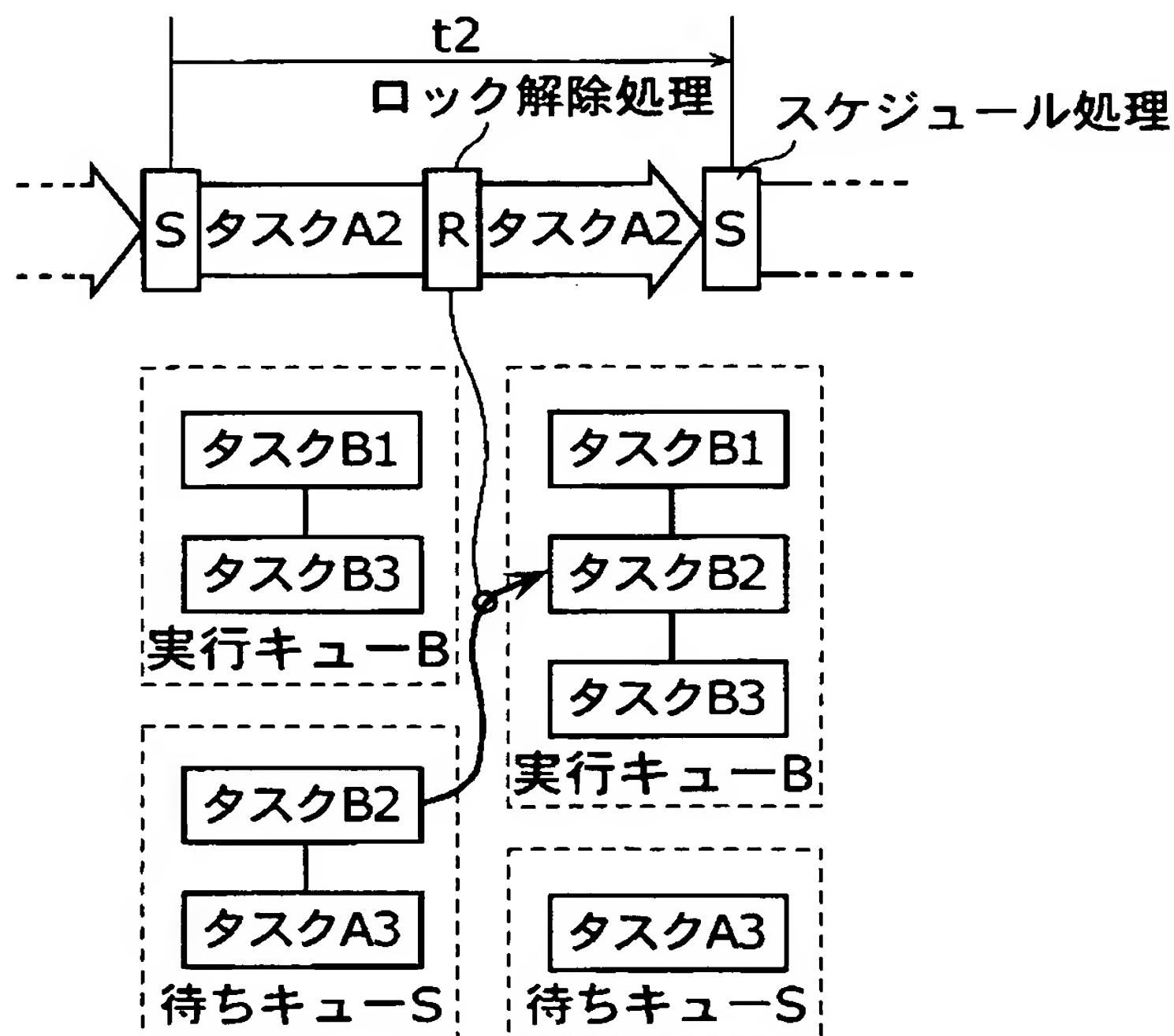
【図 7】



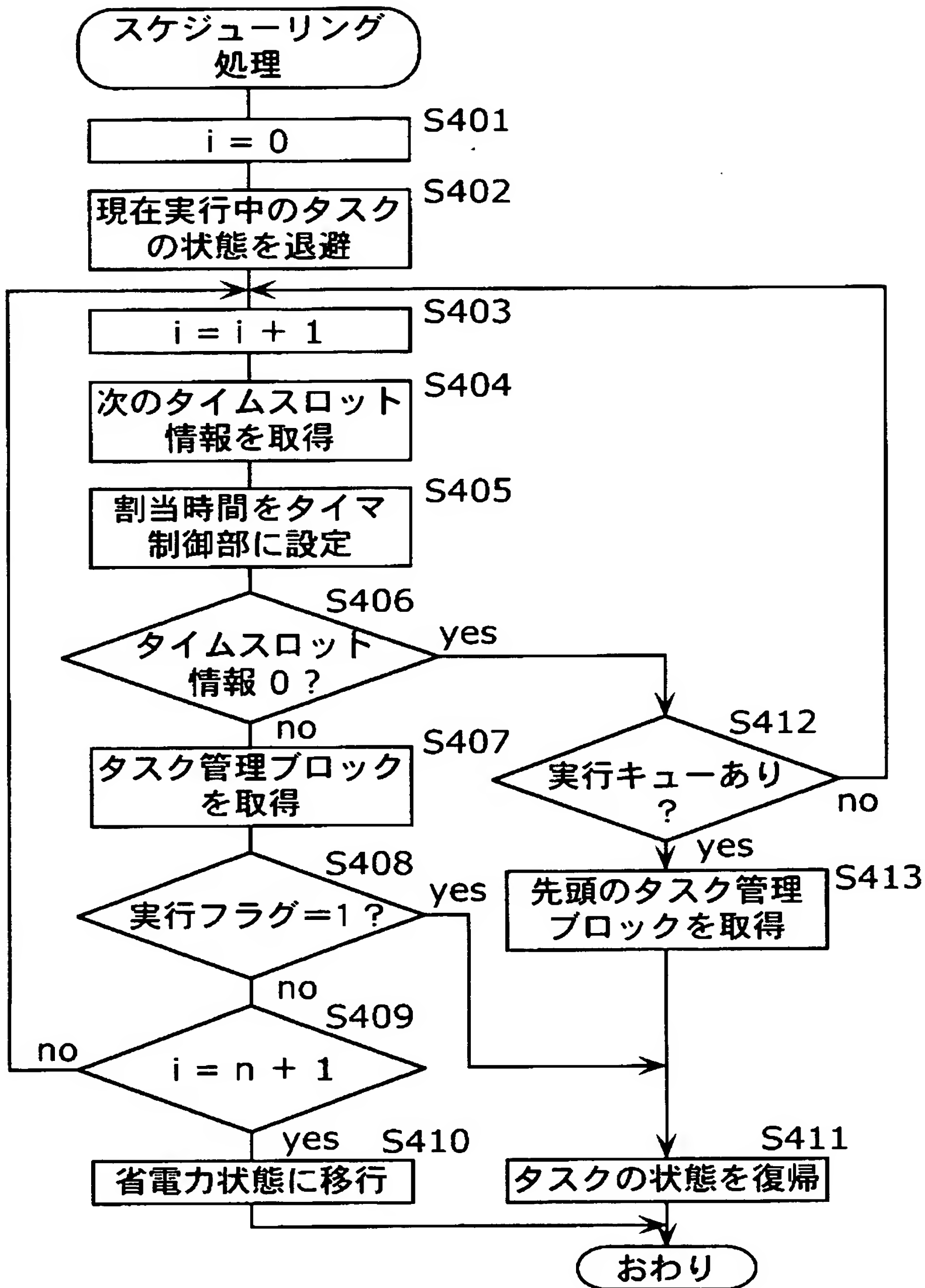
【図 8】



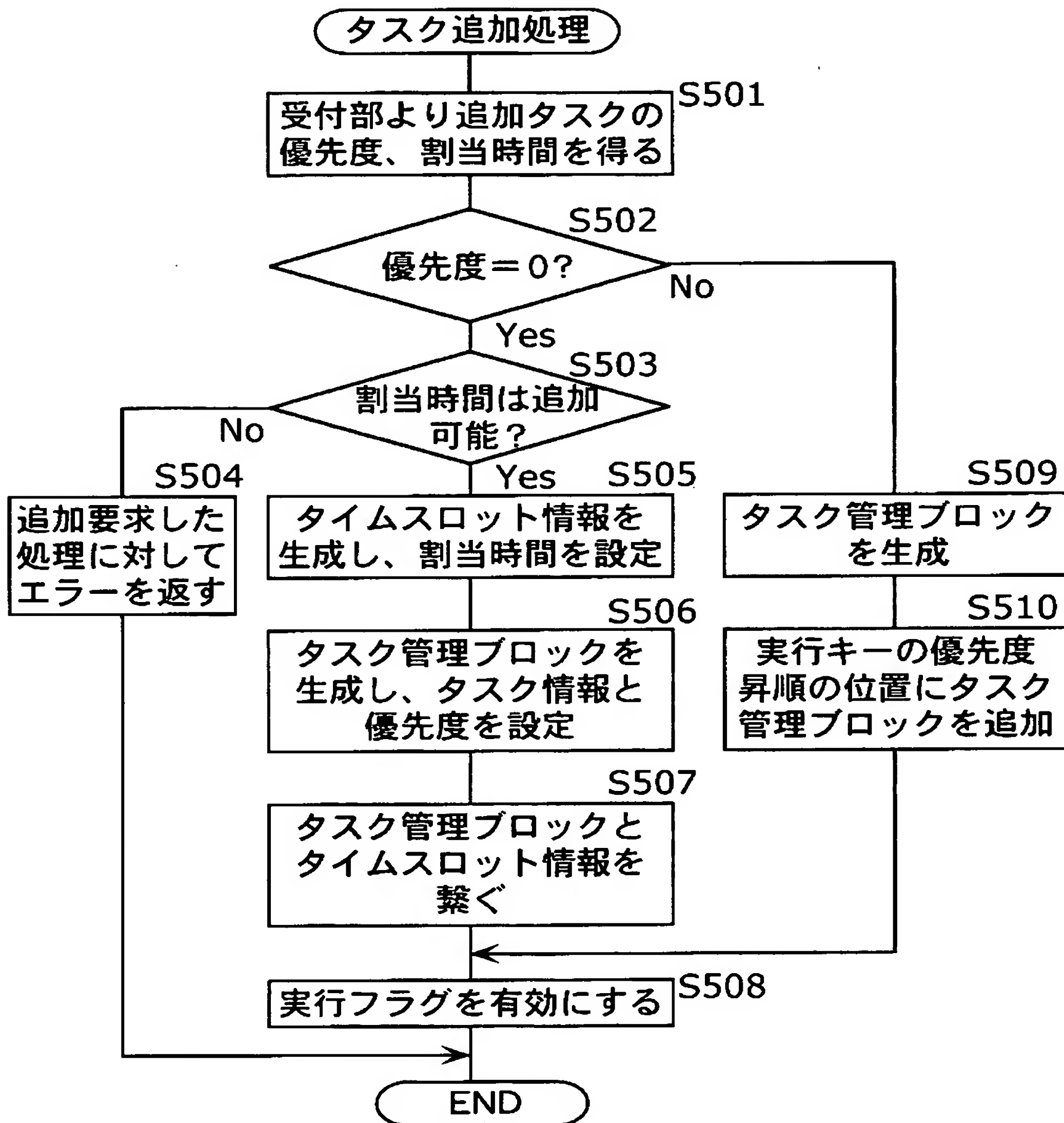
【図 9】



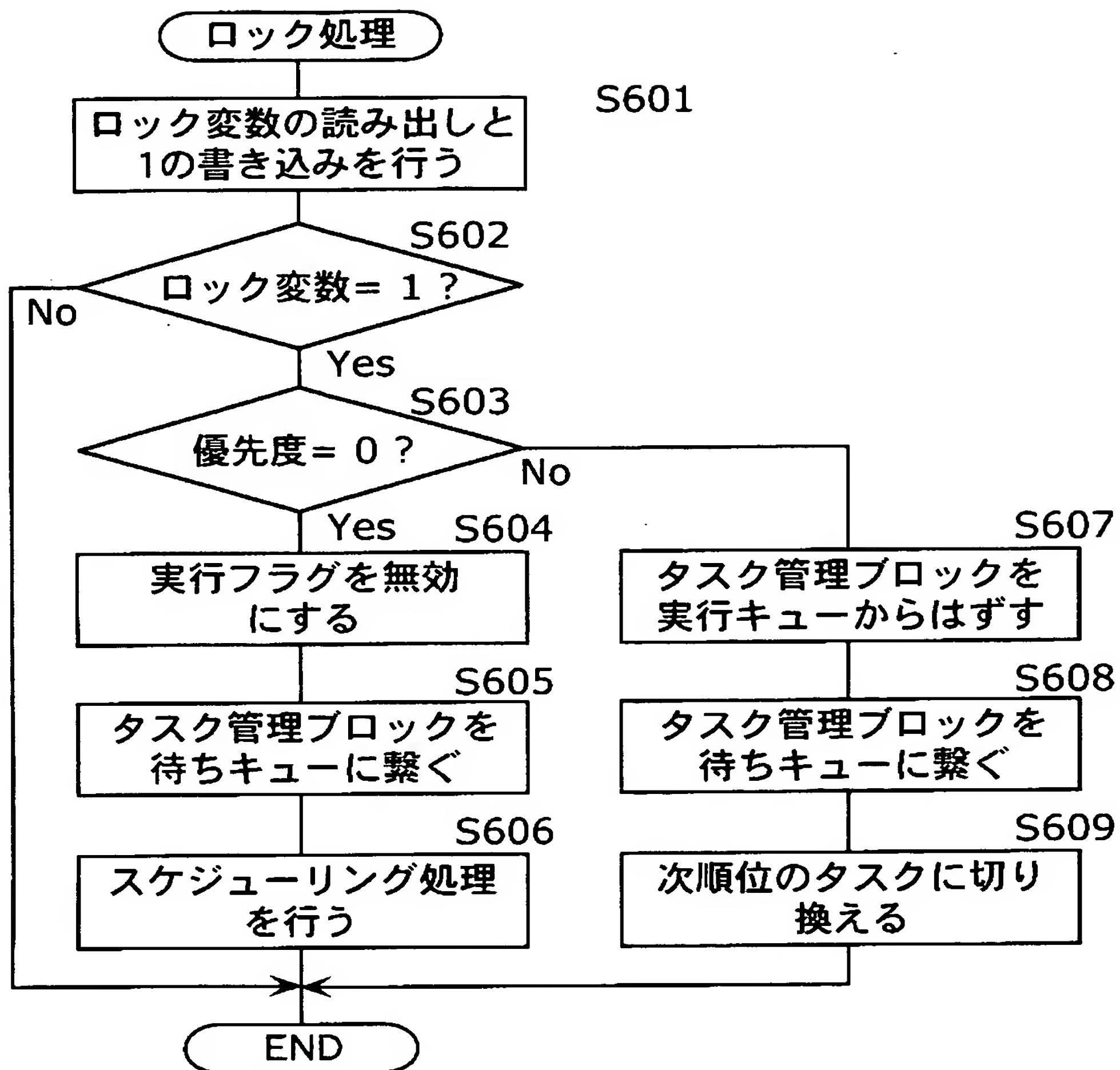
【図 10】



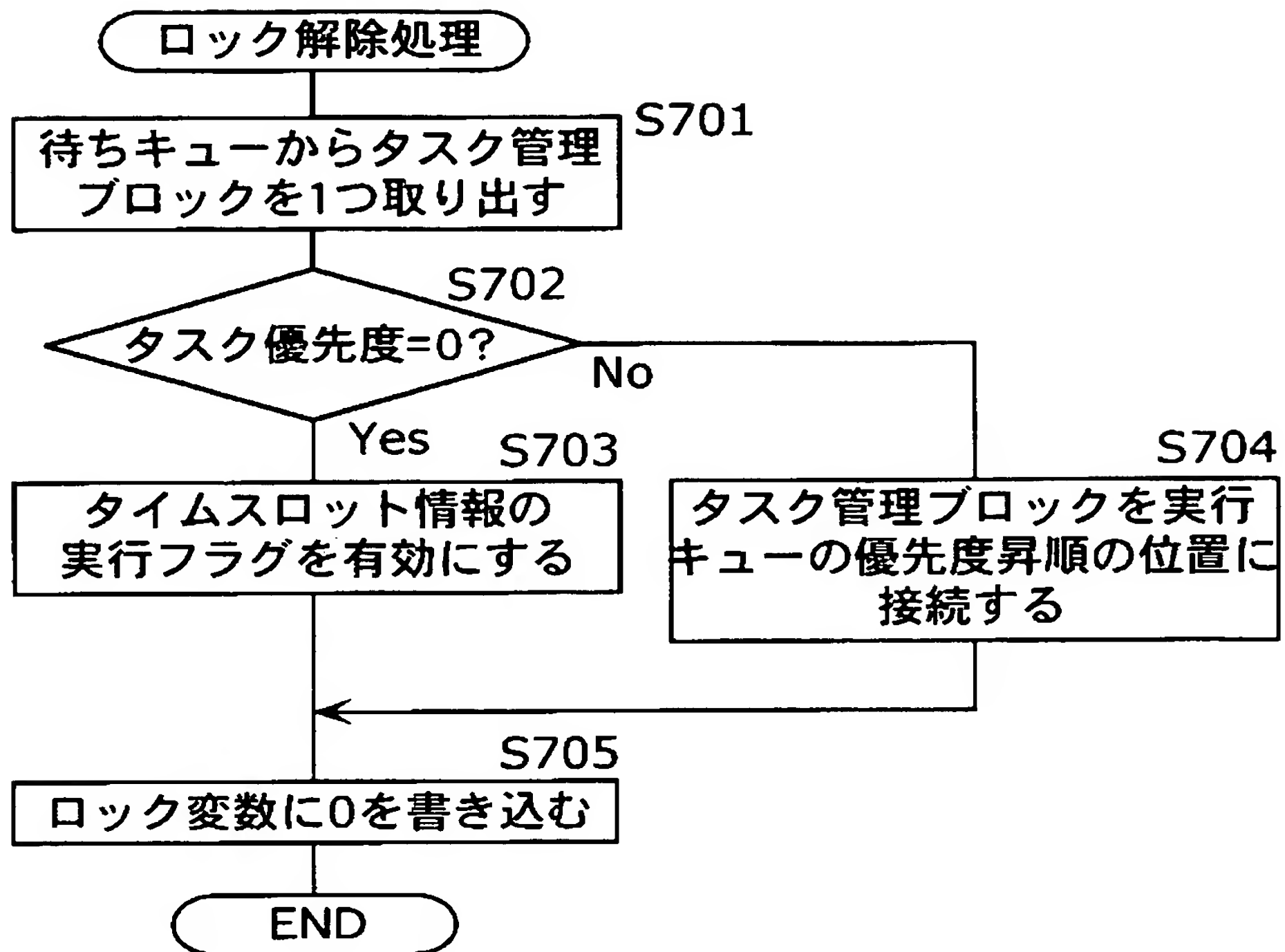
【図 11】



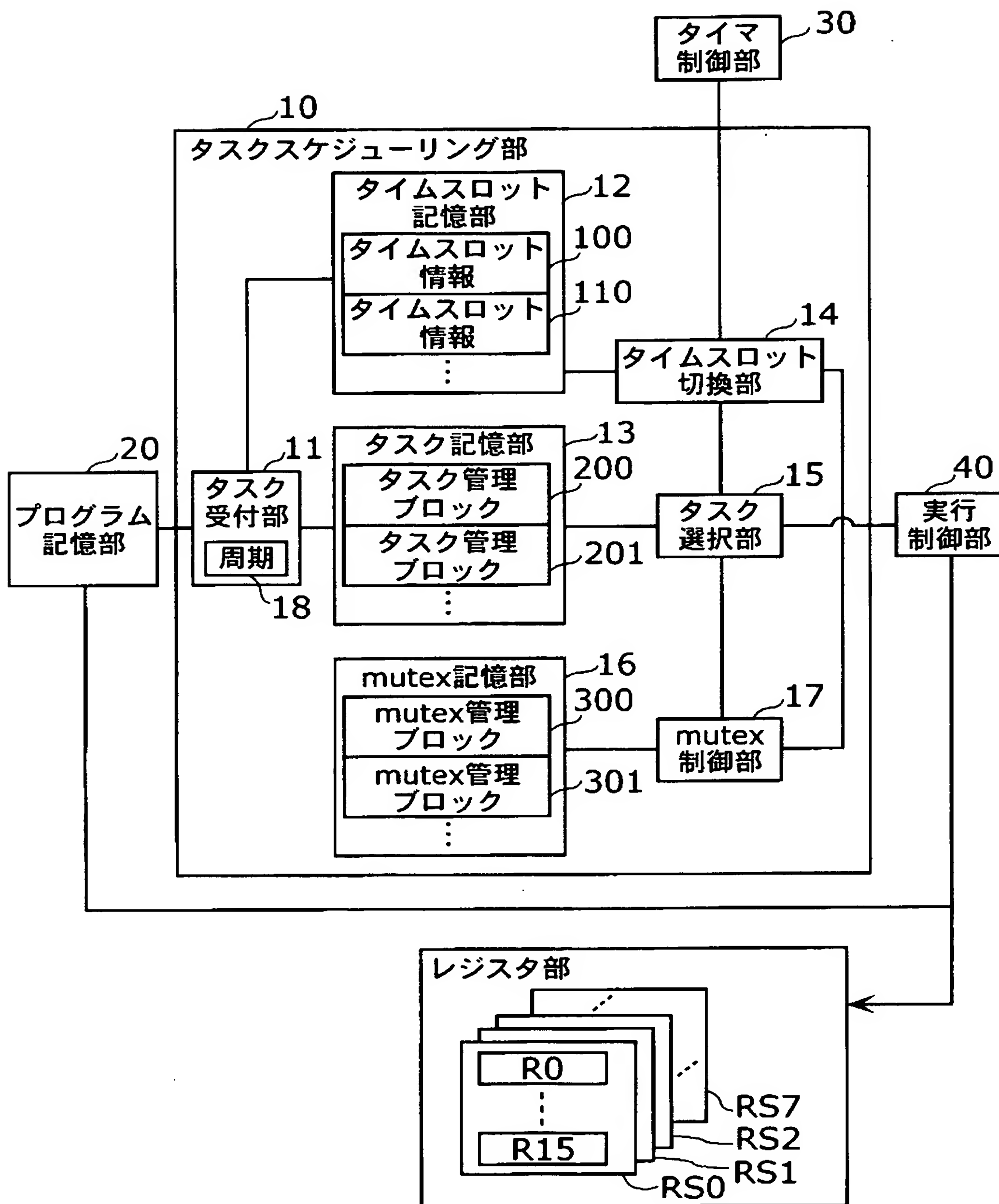
【図 12】



【図 13】



【図 14】





【書類名】 要約書

【要約】

【課題】 各タスクの必要性能を満たすための優先度あるいは割当時間を指定するプログラム設計を容易にし、プログラム変更の柔軟性を有するタスク切換装置を提供する。

【解決手段】 第1タイプのタスクのタスク管理ブロック 2 1 0、2 2 0 それぞれをタイムスロット情報 1 1 0、1 2 0 に1対1で割り当て、第2タイプの複数のタスク管理ブロック 2 0 0、2 0 1、2 0 2 を1つタイムスロット情報 1 0 0 に多対1で割り当て、タイムスロット情報 1 0 0 のタイムスロットに切り換えた場合に優先度に従って1つのタスク管理ブロックを選択し、タイムスロット情報 1 0 0 以外のタイムスロットに切り換えた場合にそれに割り当てられたタスク管理ブロックを選択し、そのタスクを実行させる。

【選択図】 図 2



認定・付加情報

特許出願の番号	特願 2 0 0 3 - 0 6 8 8 3 1
受付番号	5 0 3 0 0 4 1 6 2 4 7
書類名	特許願
担当官	第七担当上席 0 0 9 6
作成日	平成 1 5 年 3 月 1 7 日

< 認定情報・付加情報 >

【提出日】 平成15年 3月13日

次頁無

特願 2 0 0 3 - 0 6 8 8 3 1

出 願 人 履 歷 情 報

識別番号

[0 0 0 0 0 5 8 2 1]

1 . 変更年月日

1 9 9 0 年 8 月 2 8 日

[変更理由]

新規登録

住 所

大阪府門真市大字門真 1 0 0 6 番地

氏 名

松下電器産業株式会社